



**IMPLEMENTING AN INFORMATION RETRIEVAL AND  
VISUALIZATION FRAMEWORK FOR HETEROGENEOUS  
DATA TYPES**

THESIS

Andrew J. Kowalchuk, Captain, USAF

AFIT/GCS/ENG/03-09

**DEPARTMENT OF THE AIR FORCE  
AIR UNIVERSITY**

**AIR FORCE INSTITUTE OF TECHNOLOGY**

---

**Wright-Patterson Air Force Base, Ohio**

APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED.

The views expressed in this thesis are those of the author and do not reflect the official policy or position of the United States Air Force, Department of Defense or the U.S. Government.

AFIT/GCS/ENG/03-09

IMPLEMENTING AN INFORMATION RETRIEVAL AND VISUALIZATION  
FRAMEWORK FOR HETEROGENEOUS DATA TYPES

THESIS

Presented to the Faculty

Department of Electrical and Computer Engineering

Graduate School of Engineering and Management

Air Force Institute of Technology

Air University

Air Education and Training Command

In Partial Fulfillment of the Requirements for the  
Degree of Master of Science in Computer Engineering

Andrew J. Kowalchuk, BS

Captain, USAF

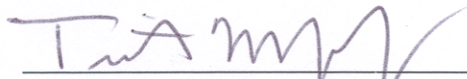
March 2003

APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED.

IMPLEMENTING AN INFORMATION RETRIEVAL AND VISUALIZATION  
FRAMEWORK FOR HETEROGENEOUS DATA TYPES

Andrew J. Kowalchuk, BS  
Captain, USAF

Approved:



Timothy M. Jacobs, PhD, Lt Col, USAF, Committee Chairman  
Department of Electrical and Computer Engineering

13 Mar 03

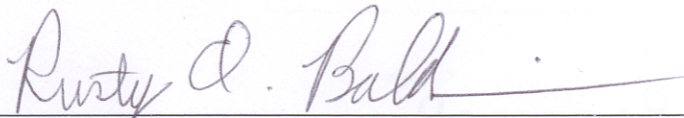
Date



Karl S. Mathias, PhD, Lt Col, USAF, Committee Member  
Department of Electrical and Computer Engineering

13 MAR 03

Date



Rusty O. Baldwin, PhD, Maj, USAF, Committee Member  
Department of Electrical and Computer Engineering

13 Mar 03

Date

## **Acknowledgments**

I would like to thank my wife for her faithful support throughout my educational experience. Her support and understanding throughout the process made this work possible.

I would also like to thank Lt Col Jacobs for the time he spent preparing me for and guiding me through this research project and my committee members, Lt Col Mathias and Maj Baldwin, for taking the time to provide constructive feedback.

Andrew J. Kowalchuk

## Table of Contents

<b>1</b>	<b>INTRODUCTION.....</b>	<b>1</b>
1.1	PROBLEM STATEMENT .....	2
1.2	CONSTRAINTS .....	2
1.3	GOALS .....	3
1.4	SUMMARY .....	5
<b>2</b>	<b>BACKGROUND MATERIAL .....</b>	<b>6</b>
2.1	INTRODUCTION .....	6
2.2	DEPARTMENT OF DEFENSE APPLICATIONS.....	6
2.2.1	<i>OODA Loop</i> .....	6
2.2.2	<i>Joint Vision 2020</i> .....	8
2.3	JOINT BATTLESPACE INFOSPHERE CONCEPT .....	10
2.4	DISTRIBUTED SYSTEMS.....	12
2.5	DATA NEEDS FOR THE JBI .....	14
2.6	QUERIES.....	14
2.7	MULTIMEDIA QUERIES.....	15
2.8	VISUALIZATION .....	16
2.9	EXISTING TECHNOLOGIES .....	18
2.9.1	<i>Jini<sup>TM</sup> Network Technology</i> .....	18
2.9.2	<i>RJBI</i> .....	19
2.9.3	<i>CoABS</i> .....	19
2.9.4	<i>Integrating Heterogeneous Systems</i> .....	21
2.9.5	<i>InfoCrystal</i> .....	21
2.9.6	<i>Bubble World</i> .....	22
2.9.7	<i>Garlic</i> .....	27
2.10	SUMMARY .....	29
<b>3</b>	<b>DESIGN .....</b>	<b>30</b>
3.1	REQUIREMENTS.....	30
3.2	DESIGN CONSIDERATIONS.....	33
3.2.1	<i>General</i> .....	33
3.2.2	<i>Middleware</i> .....	34
3.2.3	<i>Data</i> .....	36
3.3	FULFILLING REQUIREMENTS .....	36
3.4	CONCLUSION.....	46
<b>4</b>	<b>IMPLEMENTATION .....</b>	<b>47</b>
4.1	EXISTING TECHNOLOGIES .....	47
4.2	METADATA .....	47
4.3	INFORMATION RETRIEVAL AND STANDARD INTERFACE .....	49
4.4	QUERY USER INTERFACE .....	51

4.5	VISUALIZATION .....	54
4.6	QUERY TRANSLATOR.....	57
4.7	QUERY REFINEMENT.....	57
4.8	RESOURCE LIST MANAGEMENT .....	57
4.9	RESOURCE MANAGEMENT .....	58
4.10	SUMMARY .....	60
<b>5</b>	<b>RESULTS .....</b>	<b>61</b>
5.1	SYSTEM EVALUATION.....	61
5.1.1	<i>Information Retrieval</i> .....	61
5.1.2	<i>Visualization and Query User Interface</i> .....	62
5.1.3	<i>Standard Query Interface</i> .....	65
5.1.4	<i>Query Translation</i> .....	66
5.1.5	<i>Query Refinement</i> .....	66
5.1.6	<i>Resource List Management</i> .....	66
5.1.7	<i>Resource Management</i> .....	66
5.2	SYSTEM IMPACT.....	67
5.2.1	<i>Shaded Polygons</i> .....	67
5.2.2	<i>Method for Heterogeneous Queries</i> .....	68
5.3	FUTURE WORK .....	68
5.3.1	<i>Bubble World Incorporation</i> .....	68
5.3.2	<i>CoABS Integration</i> .....	69
5.3.3	<i>Better Information Retrieval</i> .....	70
5.3.4	<i>Query Translators</i> .....	70
5.3.5	<i>Query Refinement</i> .....	70
5.3.6	<i>Hierarchical Template List</i> .....	71
5.4	CONCLUSION.....	71

## List of Figures

Figure 2-1. Boyd's OODA Loop [1] .....	7
Figure 2-2. Joint Vision 2020 Overview [5].....	9
Figure 2-3. JBI Overview [9].....	12
Figure 2-4. Knowledge Crystallization [2] .....	17
Figure 2-5. How Jini <sup>TM</sup> technology works [14].....	18
Figure 2-6. InfoCrystal [11].....	22
Figure 2-7. Bubble World #1 .....	24
Figure 2-8. Bubble World #2 .....	24
Figure 2-9. Bubble World #3 .....	25
Figure 2-10. Bubble World #4 .....	27
Figure 2-11. Garlic System Architecture [3] .....	28
Figure 3-1. Plotting Example .....	40
Figure 3-2. Angle Calculation.....	41
Figure 3-3. Contribution Equations .....	41
Figure 3-4. Example Calculations.....	42
Figure 3-5. Example Summation .....	42
Figure 3-6. Plotting Error Example .....	43
Figure 4-1. Metadata Structure .....	48
Figure 4-2. Structure for Standard Query Interface .....	50
Figure 4-3. Updated Structure for Standard Query Interface .....	50
Figure 4-4. Query Wizard Screen #1 .....	52
Figure 4-5. Query Wizard Screen #2 .....	53
Figure 4-6. Query Wizard Screen #3 .....	53
Figure 4-7. Visualization Screenshot.....	54
Figure 4-8. Example of Zoom Capability .....	55
Figure 4-9. Detail of Single Data Point .....	56
Figure 4-10. Query Translator Structure.....	57
Figure 4-11. Publication Wizard Step #1.....	59
Figure 4-12. Publication Wizard Step #2.....	59
Figure 4-13. Metadata Template Creation Tool .....	60



## **Abstract**

In today's information focused world, there is no lack of entities focused on information gathering. However, there is still a widespread epidemic of information starvation in the Department of Defense (DoD). This starvation is attributed to the lack of interoperability between information gatherers and information consumers. To alleviate this problem, the DoD has put forth a vision of a Joint Battlespace Infosphere (JBI).

This research proposes a framework for sharing and finding resources in a JBI. The framework uses an extensible metadata specification, agent technology, and the Control of Agent Based Systems (CoABS). It provides several tools for publication and subscription of resources, including a visual query wizard and a visualization of the results. This framework and tools provide visual query capability for the heterogeneous resources within the JBI.

# IMPLEMENTING AN INFORMATION RETRIEVAL AND VISUALIZATION FRAMEWORK FOR HETEROGENEOUS DATA TYPES

## *1 Introduction*

In today's world, most people rely heavily on information and information technology to conduct day-to-day activities. The military is no exception to this trend. In recent years, the military has focused large amounts of time and resources on having more information than the enemy. Unfortunately, most of the efforts to gain this information employ different technologies that can not communicate or cooperate with other information gathering systems. The result is many "isolated" resources that can only be partially exploited.

To make the problem even worse, the current military doctrine focuses on joint and combined operations. Joint operations involve two or more branches of the military (i.e. Army, Navy, Air Force) while combined operations involve militaries from two or more countries. This era of military cooperation has brought with it the need to share information between many more "isolated" systems.

To address this problem, the Department of Defense has created the Joint Battlespace Infosphere (JBI) concept. This concept envisions all military information sources being linked together in one environment to get the right information to the war fighter when needed. This concept brings with it many problems that need to be solved before it can come to fruition. A few of these problems are legacy system integration, querying of heterogeneous data sources, managing all available resources, and communication.

## **1.1 Problem Statement**

The purpose of the research described in this paper is to design and test a query system that aids the Department of Defense (DoD) in effectively implementing its concept of the Joint Battlespace Infosphere. The JBI concept is mainly focused on allowing many systems from all branches of the military to interact with each other in a common environment [9]. This common environment allows collaboration of effort, thus providing more effective operations in simulated and real-world environments.

Although this is a very narrow focus, the research in this paper can be applied to a much bigger problem. The essence of the JBI concept is to create interaction between numerous heterogeneous systems in a distributed environment. One key obstacle that currently stands in the way of this concept is the difficulty in finding needed resources in such a system. This is one of the main focuses of this research. The other main focus is displaying the results of such a query in a usable and informative output. Each of these main focus areas brings with it several challenges that are discussed in detail in this paper.

## **1.2 Constraints**

At the heart of these focus areas there are three underlying constraints that complicate the problem further. The first complication comes from the fact that the system is distributed in many different locations and on many different types of platforms. The second complication comes from the fact that the data domain of the JBI concept spans a virtually limitless set. Therefore, very few assumptions can be made about the resources contained in the JBI. The third constraint is that this system needs to be able to effectively handle the integration of legacy systems. This is a very important

part of the JBI concept, since much of the functionality desired in a JBI system is already contained in various legacy systems. This provides functionality to the JBI concept with minimal “cost” of development. All of these constraints have to be taken into consideration in any proposed solution. If they are not accounted for, the solution will not be able to truly satisfy the JBI concept.

### **1.3 Goals**

The ultimate goal of this research is to determine the best design for querying the resources contained in a system that implements the JBI concept and to display these results in a graphical manner that provides the best user comprehension. This overarching goal can be broken down into several smaller goals that need to be satisfied.

- Information Retrieval – Since users and applications need to find and access resources contained in the JBI, it is necessary that Information Retrieval (IR) be effectively implemented. The IR model that is implemented needs to be able to account for the numerous and heterogeneous data types that are present in the JBI at any given time. Since data types are not constrained in the JBI, it is necessary to ensure that the implemented IR model is extensible.
- Visualization – Since the system is available directly to users, query results need to be displayed in a graphical manner that aids in human understanding. Help should also be provided to the user to aid in selecting the correct result once the query is processed.
- Query User Interface – Along the same lines as the visualization requirement, the system needs to implement an easy-to-use query user interface. This needs to aid the user in posing a query to the system. Based on the constraints discussed in

Section 1.2, this interface must not place undue limitations on the type and number of resources contained in the system.

- Standard Query Interface – To keep from placing limitations on the system, the query functionality should be available for automated use by other parts of the system. This requires that a standard interface be developed to implement the IR model of the system.
- Query Translation – Another requirement for this system is the need to support legacy systems, including legacy query tools. Therefore, a means of translating queries into the IR model implemented in this system is required. To ensure maximum flexibility and minimal limitation on the system, the implementation of this concept needs to be extensible. This allows for more query translators to be added in the future without degrading performance or requiring major system modifications.
- Query Refinement – Due to the possibly large and ill-defined data set that this system can contain, it is necessary to aid the user in refining his query. This query refinement should provide assistance to help a wide range of users, from novice to expert.
- Resource List Management – In order to implement the above requirements, the system needs to maintain and manage a list of all resources available in the system. This list is required to make the system cognitive of what resources are available when a query is posed. Without this cognition, the system can not fulfill a query for resources.

- Resource Management – There must be a well-defined process to add or delete resources in this system. This process must comply with all implemented constraints. The process should also have a user interface associated with it, to aid users in managing their resources.

These goals lay out the minimum framework necessary to provide the query support for the JBI concept. They also provide the criteria by which the resulting system is tested against. If the system meets these goals, it proves that this concept can be achieved and thus is successful.

#### **1.4 Summary**

The ability to meet the goals set forth in this chapter provides a key component necessary to implement the JBI concept. Without a system to perform these tasks, the heterogeneous environment cannot be brought together into a single infosphere. The next chapter of this paper provides an in-depth look at the issues facing this implementation as well as some current techniques that are available to help solve these problems.

## 2 *Background Material*

### **2.1 Introduction**

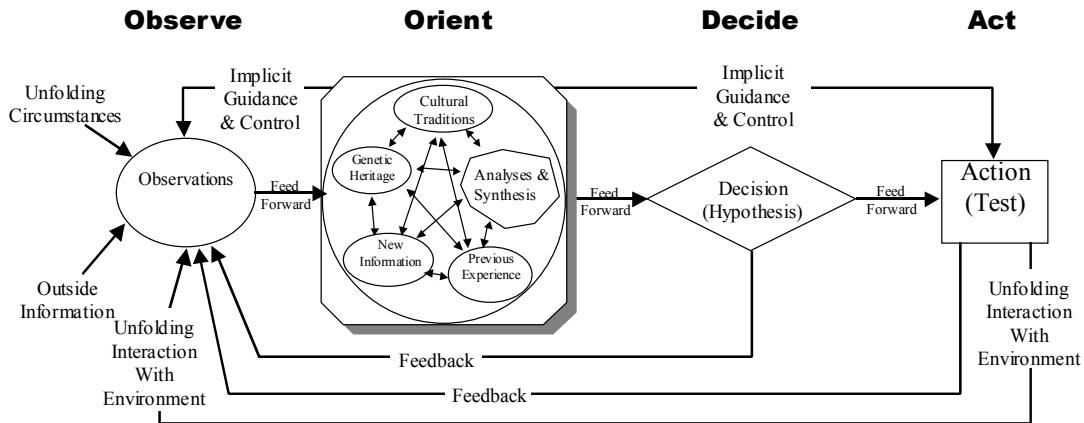
To better understand what is involved in this problem, it is necessary to learn more about the problem domain. It is also necessary to determine what solutions already exist and evaluate the suitability of each solution based on the goals set forth in Section 1.3. Once each solution is evaluated, it is possible to determine if any part or whole solution helps in this particular problem. If reuse is possible and beneficial, the amount of new components is reduced making the problem easier to solve.

### **2.2 Department of Defense Applications**

To fully understand what this problem involves, it is first necessary to explore the problem domain. The problem domain supports the decision making process of the DoD. The United States military doctrine has changed dramatically throughout its existence. The face of warfare has driven this change in doctrine. In more modern times, warfare has become markedly more dependent upon knowledge and decision-making. This has been both supported and driven by technological advances.

#### **2.2.1 OODA Loop**

In June 1995, Colonel John R. Boyd, gave a briefing entitled “The Essence of Winning and Losing” that summarized the current military doctrine as a loop consisting of four steps [1]. The four steps of this loop, as depicted in Figure 2-1, are observe, orient, decide, and act, lending to its name, the OODA Loop.



**Figure 2-1. Boyd's OODA Loop [1]**

- **Observe** – This first step occurs while the decision makers gather information relevant to the battle. The amount and quality of information gathered must be weighed with the time it takes to collect the information. Both speed and accuracy are important in this stage.
- **Orient** – In this step, the decision maker must familiarize himself with the surrounding environment, information about the enemy, and any other relevant information. This “situational awareness” gives the decision maker the picture necessary to make informed decisions.
- **Decide** – The decision maker, at this point, should have the information necessary to make an informed decision about the proper course of action. This decision is crucial in the process, as it defines the direction taken by the military forces.
- **Act** – In this part of the loop, the decision is carried out. This is the culmination of the process and is ultimately what projects the force on the enemy. The results of this step are usually dependent on the quality and timeliness of the first three



steps. The results of this step are used as feedback into the Observe step of the next decision.

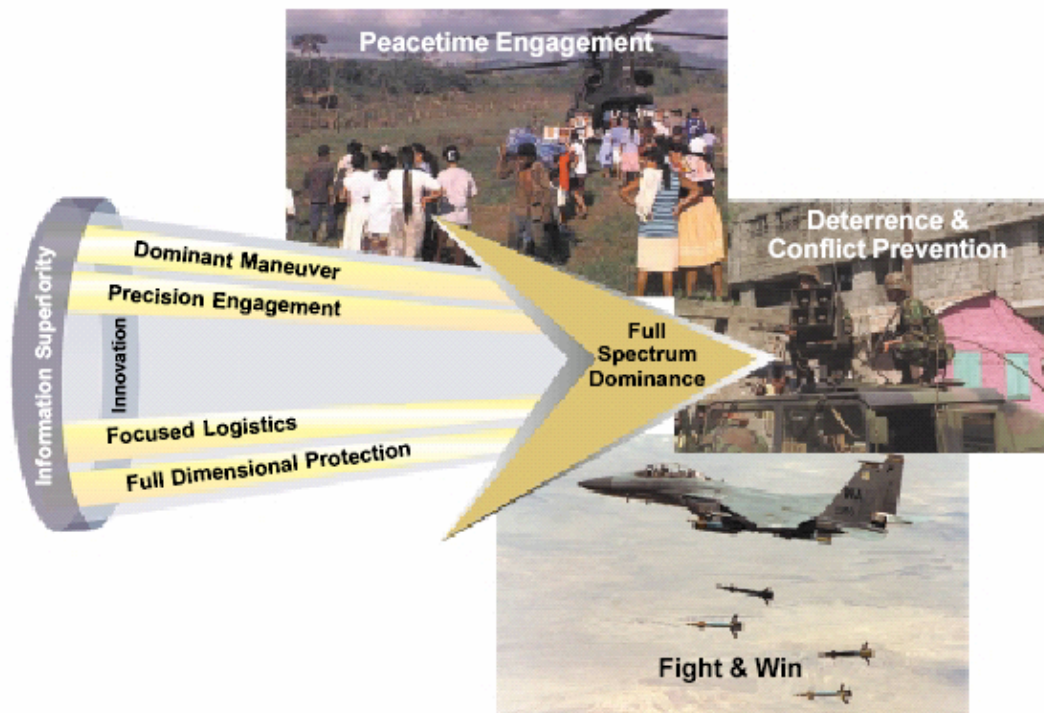
Colonel Boyd claims that all decisions go through these steps before they are carried out. By looking at the decision process like this, it is possible to understand what is necessary to get the upper hand in a battle of decisions. By fully understanding this loop, a commander is able to operate more efficiently and hopefully defeat the enemy.

Colonel Boyd's main thrust behind the OODA Loop is that by knowing how the process works, it is possible to decrease the amount of time it takes to execute the loop. If a commander can execute his entire OODA Loop faster and more accurately than his enemy, the commander effectively breaks his enemy's OODA Loop and thus gains the upper hand. Colonel Boyd goes on to show that the current state of information and its role in military operations make the OODA Loop more important than ever. The Information Age has dramatically reduced the time in which it takes to execute an entire loop. Now decisions can sometimes be made in a matter of hours or even minutes. As information resources grow and become faster, many expect the OODA Loop to be accomplished in the matter of seconds in the not so distant future.

### **2.2.2 Joint Vision 2020**

Colonel Boyd's theory of decision-making has been widely accepted by military leaders and currently drives most doctrine changes in today's military. Doctrines are now formed to make faster and more accurate decisions. The Chairman of the Joint Chiefs of Staff also followed this theory when he released *Joint Vision 2020*, a document that lays out the United States' doctrine on how to fight and win a war in the year 2020 [5]. This document is the second such document (*Joint Vision 2010* was the first) that has been

focused specifically on the future of warfare and “its purpose is to describe in broad terms the human talent – the professional, well-trained, and ready force – and operational capabilities that will be required for the joint force to succeed across the full range of military operations and accomplish its mission in 2020 and beyond” [5].



**Figure 2-2. Joint Vision 2020 Overview [5]**

As is depicted in Figure 2-2, Information Superiority is recognized as a key enabler to achieving the CJCS vision of Full Spectrum Dominance. In the following excerpt from Joint Vision 2020, this point is addressed:

Information, information processing, and communications networks are at the core of every military activity. Throughout history, military leaders have regarded information superiority as a key enabler of victory. However, the ongoing “information revolution” is creating not only a quantitative, but also a qualitative change in the information environment that by 2020 will result in profound changes in the conduct of military operations. In fact, advances in information capabilities are proceeding so rapidly that there is a risk of outstripping our ability

to capture ideas, formulate operational concepts, and develop the capacity to assess results. While the goal of achieving information superiority will not change, the nature, scope, and “rules” of the quest are changing radically. [5]

In this excerpt it is easy to see the emphasis placed on information gathering (Observe) and information processing (Orient). All of this leads to better decisions by commanders (Decide) and ultimately Full Spectrum Dominance (Act). Since technology is increasing at such a rapid rate, it is only logical to focus on these areas as the foundation for the rest of the process. If these processes can be completed faster and more accurately, the entire OODA Loop of war planning gains efficiency.

Following suit, each of the service components published its own document describing how it would train, equip, and operate to support the CJCS *Joint Vision 2020*. Each of these documents also focuses on the use of information to further increase military strength.

### **2.3 Joint Battlespace Infosphere Concept**

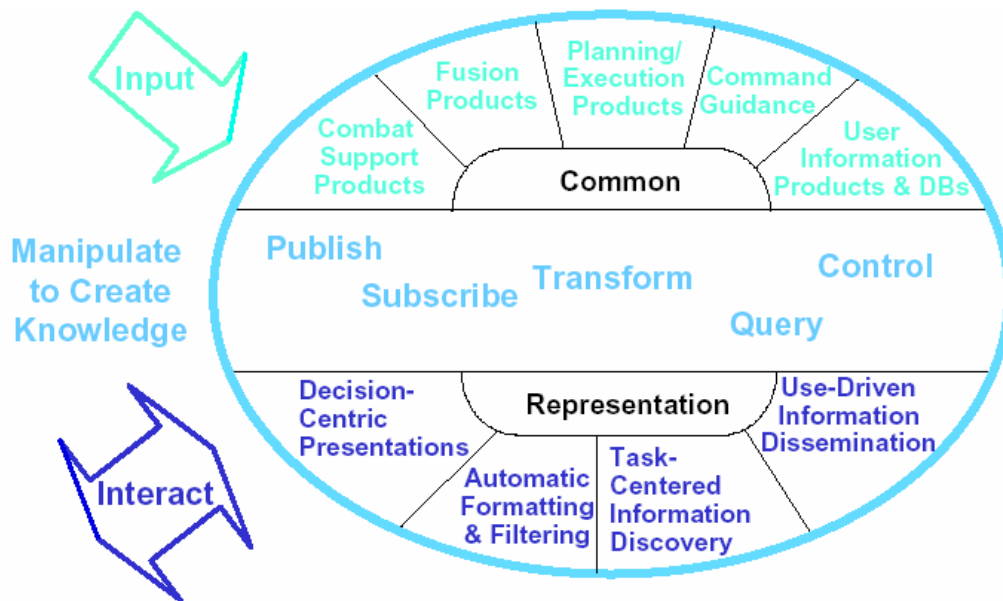
Based on the doctrine set forth in *Joint Vision 2020*, a concept arose that espouses multiple information sources that can be treated as a single information repository for all military operations. This repository is called the Joint Battlespace Infosphere (JBI). This JBI concept was initiated by the Air Force Scientific Advisory Board (AFSAB) to fulfill the goal of Information Superiority set out in *Joint Vision 2020* through a network-centric approach. According to the AFSAB:

Network-centric warfare is a first step in the direction of forming a common view of the battlespace by ensuring ubiquitous connectivity. Network-centric systems gain their operational advantage by integrating existing planning and warfighting systems via a communications network. The BI extends the concept of the network-centric system. It remains essential that existing and evolving function-specific systems be interconnected and able to intercommunicate. But in the BI, capabilities for intelligent data transformation, information exchange, knowledge sharing, and processing provide the operational advantage. [9]

The BI provides a highly tailored repository of, or access to, information that is designed to support a particular geographic area or mission. The intent of the BI is to have a 'single place,' a 'virtual system of information systems,' that serves as a clearinghouse and a workspace for anyone contributing to the accomplishment of the operation -- for example, weather, intelligence, logistics, or personnel. The use of the BI seamlessly integrates multiple sources of data, enables automated manipulation of data, provides faster response time, and produces tailored information to support warfighter decision making throughout all functional staff activities. [9]

From this description of the JBI, it is easy to see that it is to be an aid “to support warfighter decision making” [9], which is in line with both Joint Vision 2020 and Colonel Boyd’s OODA Loop. Since the JBI is only a concept, there needs to be an infrastructure that supports operations that contribute to this concept. In order to implement this concept, the infrastructure needs to be able to perform three main operations. In Figure 2-3, these operations are depicted outside of the oval. First, it must allow resources to be added to the repository. Second, it must manage the content of the repository in a useful manner. Thirdly, it must support the use and manipulation of the resources by authorized users. These requirements are a minimal set of operations that need to be provided by an infrastructure that implements the JBI concept. When implemented, these requirements facilitate the operations found inside the oval. These operations (Publish, Subscribe, Transform, Query, and Control) help fuse the information sources into a single picture for the commander.

These three operations (input, management, and manipulation) describe the minimum functionality necessary to implement the JBI concept. However, there are also some constraints that can be gleaned from the preceding definition of the JBI. Although not stated explicitly, it is implied that the JBI needs to be able to function in a distributed



**Figure 2-3. JBI Overview [9]**

environment. This is evident in the description where it discusses the need for “integrating existing planning and warfighting systems via a communications network” [9]. The next constraint placed on the infrastructure is that it needs to be able to handle “multiple sources of data” [9]. This means that the system not only needs to handle multiple input sources for data, it must also handle heterogeneous data types. These constraints each provide complications to the implementation of the JBI concept and are discussed further in Sections 2.4 - 2.6.

## **2.4 Distributed Systems**

Since the JBI concept needs to be implemented in a distributed environment, there are several distributed system concerns. These concerns include issues with naming, location independence, locating entities, reliability, and consistency. The concepts discussed in this section are taken from

Distributed Systems: Principles and Paradigms. [15]

A key concept in all computer systems, but that takes on new significance in distributed systems, is naming. “An important concept with naming is that a name can be resolved to the entity it refers to” [15]. This is the main purpose in naming entities within a system. At first, naming does not seem to be a significant problem in an implementation of the JBI concept. However, since the implementation needs to be distributed, the naming system must also be distributed. This brings on added concerns of controlling and defining the naming system. Some part of this distributed system needs to ensure that duplicate names are not improperly used.

Another concern with some distributed systems is the need for location independence. This means that the physical location of an entity is not important for location and use. This is definitely an important concept due to the propensity of the military to change locations rapidly. This concern requires that the naming convention not be tied to physical or even logical location. In addition, there must be some means to dynamically associate names with locations. This system ensures that stagnate links are not maintained and that moving entities are always reachable.

This concern also bleeds over into the next concern of locating mobile entities. Since mobility is a major aspect of the JBI concept, the infrastructure needs to be able to locate and effectively communicate with entities even if they are not always located in the same place. This becomes especially difficult if an entity needs to move while being used. Current communications need to be rerouted to the new location of the entity.

The last two concerns, consistency and reliability, are somewhat related in that they contribute to the integrity and completeness of the system. Consistency ensures that a single entity is viewed the same by all other entities at a point in time. This objective

helps ensure that the correct decisions are made. Reliability is concerned with the availability of all information. It seeks to incorporate all available resources and ensure availability.

These concerns of distributed systems play a large factor in any implementation of the JBI concept and drive the design of such an implementation.

## **2.5 Data Needs for the JBI**

Another key issue concerning any JBI implementation is the broad range of data and resources that can be a part of the JBI. In the excerpt in Section 2.3, it is noted that the JBI concept is designed to serve “as a clearinghouse and a workspace for anyone contributing to the accomplishment of the operation -- for example, weather, intelligence, logistics, or personnel” [9]. This implies that there can be many data types contained within the supporting infrastructure. These types might include images (e.g. intelligence photos), documents (e.g. weather reports), simulation objects, and many others.

In addition, new types may be introduced at any time and need to be integrated into the system. Since the JBI concept is derived from a vision for military operations in the future, it is necessary to be able to adapt the system as new technology and ideas are born. This means providing maximum flexibility to any infrastructure that implements the JBI concept.

## **2.6 Queries**

In this modern day of information dependence, systems commonly become too large or contain more information than a human can possibly organize. Queries make finding particular bits of information in large repositories possible. Due to the increased emphasis placed on information by Joint Vision 2020, it is understandable that the JBI

concept is expected to maintain large amounts of information. However, there are several key attributes of the JBI repository that pose complications when considering query techniques.

One of the most prevalent query domains that exists in modern applications is searching the World Wide Web (WWW). While the WWW contains enormous amounts of information (probably more than any JBI implementation would need), all of the resources are treated as textual documents and searched accordingly. This allows for simpler query standards.

Another common query application is found in relational databases. Queries of this kind can be concerned with multiple data types (text, number, dates, etc.), even at the same time. However, the queries (and the data for that matter) are constrained by the types that are allowed by the database. This is usually a fairly small set.

## **2.7 Multimedia Queries**

A shortcoming of most traditional query systems is that each can only compare one type at a time. Therefore, textual documents cannot directly be compared to image documents. Great advances have been made in rich query standards for many single types. For example, systems have been created that can query a set of images based on shapes, colors, patterns, and other similarities. While these queries provide a very rich query language and fairly accurate results, they are highly specialized for a certain data type. The algorithm and techniques used to query images can not be applied to a group of textual documents.

A common solution to this problem is the use of metadata. Metadata is defined as “data about data” [19]. Metadata provides a common ground for the different types



participating in a collection. The same metadata attributes can be filled in for a textual document as with an image. Since the metadata formats are the same, the two documents (text and image) now share some common ground and can be queried. However, significant precision is lost in this conversion to metadata.

Despite this loss of precision, metadata is still the most common method to query multiple data types. To minimize the loss, it is necessary to follow standard guidelines when formulating metadata for objects. These standards can vary between systems, but should always be adhered to within any one system. Some work has been done by the Dublin Core Metadata Initiative (DCMI) to determine some generic guidelines for metadata. One of their recommendations is that metadata be limited to the smallest set of information that fully describes an object [6]. This sometimes can be a difficult task, especially when the data domain is not strongly constrained.

One of the main focuses of DCMI was to develop a standard set of metadata that describes all objects. The result is a list of 15 metadata attributes that can be associated with any data and serves to distinguish that data from all other data. The reason this was developed was to support interoperability between systems. If all systems implemented this metadata standard, the systems could share resources.

## **2.8 Visualization**

The ultimate goal of this research is to produce a query visualization system. Therefore, it is necessary to explore a few of the concepts of information visualization that impact this research. The overall goal of any visualization is to convey an idea or message to the user. The process by which information is gathered and processed by a person is the knowledge crystallization process [2]. An effective visualization speeds the

user's knowledge crystallization process, depicted in Figure 2-4. This process is performed when users begin a search for data. The purpose of a visualization is to streamline this process and make it easier and quicker for the user to find the information he needs [2].

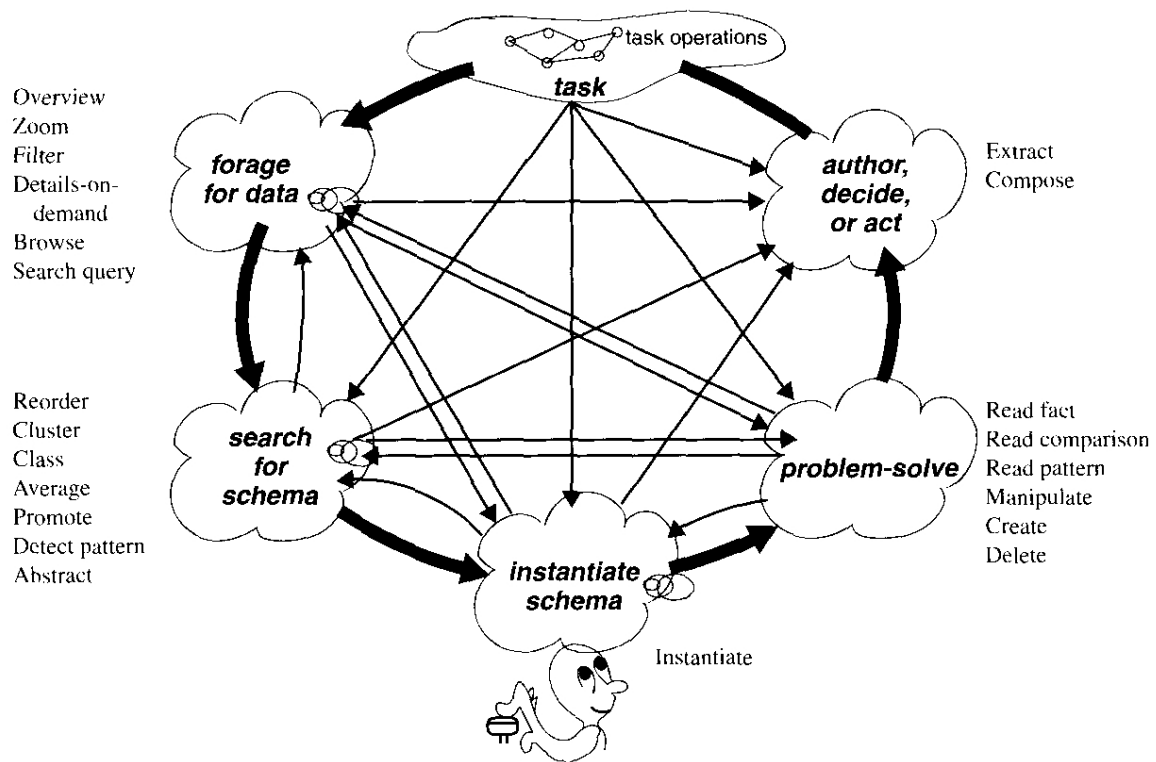


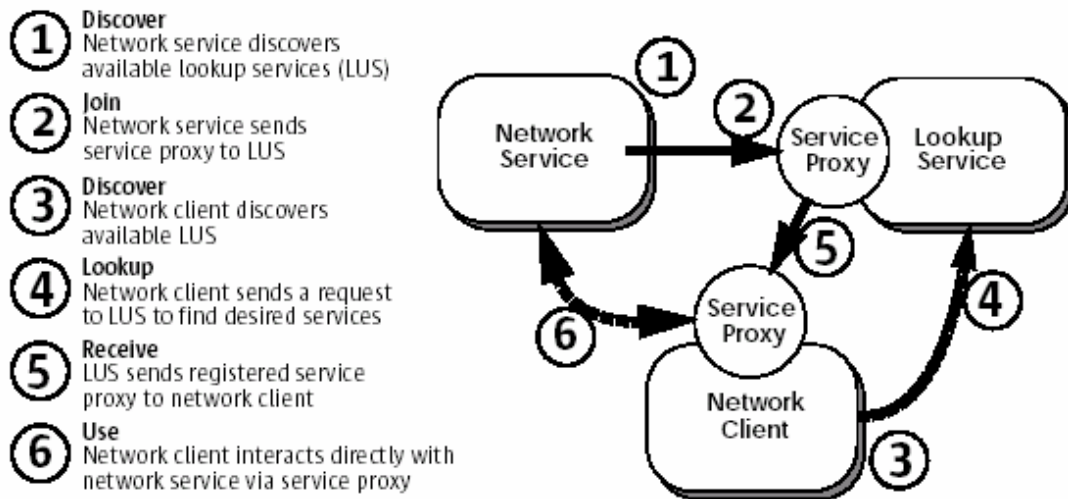
Figure 2-4. Knowledge Crystallization [2]

Visualizations use concepts such as color, shading, motion, and placement to aid in a user's understanding of data. However, just as these concepts can aid in the process, they can also detract from the process. An effective visualization maximizes the benefit and minimizes the detriment of all of the visual stimuli that make up the visualization.

## 2.9 Existing Technologies

### 2.9.1 Jini™ Network Technology

Jini™ Network Technology is a middleware solution that “consists of an infrastructure and programming model that addresses the fundamental issues of how clients and services discover and connect with each other to form an impromptu community” [14]. Jini™ uses six steps to implement this process, as seen in Figure 2-5.



**Figure 2-5. How Jini™ technology works [14]**

The first step of the process is *discovery*. In this step, a new service discovers all available lookup services. A lookup service is “a directory or index of available services, where proxies to these services and their code are stored” [14]. Once the service has located all available lookup services, it *joins* the community by registering its proxy with the lookup services. A client initiates Step 3, also called discovery. Once the client is aware of the lookup services available, it sends a *lookup* request to the lookup service. In Step 5, the lookup service then returns a proxy for a matching service. Finally, the client

communicates with the requested service via the service proxy that was returned by the lookup service.

This process provides a flexible, yet powerful, infrastructure that facilitates locating and communicating with other services. One of the most important features of this process is the concept of leasing. In Figure 2-5, Step 2, the network service is only given a lease when joining the community. This lease has an expiration time that triggers removal of that service from the community if the lease runs out. However, prior to the lease termination time, the service can negotiate a new lease with the lookup service [14]. This controls the problem of stagnant links due to lack of deregistering or premature termination of services.

Jini<sup>TM</sup> Network Technology provides a powerful infrastructure that allows services to interact with little foreknowledge of location or underlying network. It offers a highly scalable solution to the problems of network transport.

### **2.9.2 RJBI**

RJBI is an implementation of the JBI concept built on top of the Jini<sup>TM</sup> Network Technology. The RJBI, released by the Air Force Research Laboratory, provides a simple publish and subscribe infrastructure. The functionality that is added to Jini<sup>TM</sup> is very minimal.

### **2.9.3 CoABS**

CoABS, Control of Agent Based Systems, is another framework built on top of the Jini<sup>TM</sup> Network Technology [22]. Unlike the RJBI, CoABS provides much more support on top of the Jini<sup>TM</sup> infrastructure. CoABS, as its name implies, is designed to

support the seamless integration of agent-based systems. Like Jini™ CoABS provides a scalable and flexible environment for systems to participate in.

To fully understand the environment, it is first necessary to cover a few key issues concerning agents and agent-based systems. Agents are defined as "an encapsulated computer system that is situated in some environment and that is capable of flexible, autonomous action in that environment in order to meet its design objectives" [7]. These agents are designed to independently locate and negotiate communications with other agents in the environment to accomplish their assigned objectives.

Agents use CoABS to register themselves and to gain access to several services that help them communicate. One of these services is a lookup service. CoABS uses the Jini™ lookup service to implement this function. This lookup service aids the agent in finding other agents that it can communicate with to achieve its goal.

Another service offered by CoABS is agent messaging. This messaging service provides a transport mechanism to deliver messages between agents. This is done through the use of agent representatives and registration helpers. This messaging service also aids the agents in determining if their native communications are compatible.

Finally, CoABS offers a Graphical User Interface that allows an administrator to manage and monitor the CoABS Grid. This grid is the combination of Hypertext Transfer Protocol (HTTP) daemons, Lookup Service (LUS) daemons, and Remote Method Invocation (RMI) daemons. These daemons provide the services necessary for agents to advertise their capabilities and solicit the capabilities of other agents.

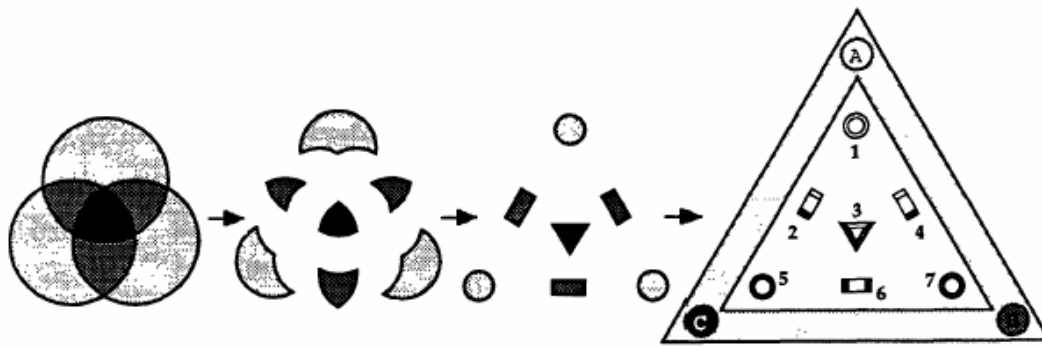
#### **2.9.4 Integrating Heterogeneous Systems**

Since this research is part of a bigger concept, it is necessary to explore the other parts in development to determine if there are any additional requirements or constraints. Another part of the problem that is currently being researched is the “Data Framework for Integrating Heterogeneous Systems Using Agents, XML, and CoABS” by Rick Roell [10]. As the name implies, this research is concerned primarily with the use of agents to integrate heterogeneous systems. The focus of his research is to define a standard interface that can accommodate communications between many different systems. The research focuses on the use of agent wrappers to incorporate legacy data sources and legacy systems into the JBI concept.

In his paper, Roell weighs the advantages of an object-oriented approach versus an agent-based approach [10]. His final analysis shows that agents have a distinct advantage over objects in implementing the JBI concept [10]. Due to this finding, the research presented in this paper will also be constrained to the use of an agent-based system.

#### **2.9.5 InfoCrystal**

InfoCrystal is a product of the research of Anselm Spoerri from the Massachusetts Institute of Technology [11]. InfoCrystal is a query visualization tool that aids the user in selecting appropriate matches when multiple query terms are used. The basic concept behind InfoCrystal begins with the Venn diagram. In a Venn diagram of three circles, there are seven distinct areas that can be extracted. Each of these areas is then given a location and glyph representation in a triangular grid. This process is depicted in Figure 2-6.



**Figure 1:** shows how we can transform a Venn diagram into an iconic display, called the *InfoCrystal*, which represents all the possible Boolean queries involving its inputs in a normal form (see section 2.2). The interior icons have the following Boolean meanings: 1 = (A and (not (B or C)), 2 = (A and C and (not B)), 3 = (A and B and C), 4 = (A and B and (not C)), 5 = (C and (not (A or B))), 6 = (B and C and (not A)), 7 = (B and (not (A or C))).

**Figure 2-6. InfoCrystal [11]**

It is important to note that the shape of the glyph depicts how many terms that grouping matches. Each of the glyphs is a grouping of relevant documents. The number of documents in each grouping is denoted right next to the glyph. This model provides an intuitive way to easily identify the most relevant group of documents in a collection. However, InfoCrystal is heavily reliant on the existence of a Boolean Query for the data in question. If no such query exists, or is not desired, than this method cannot be used to plot the query.

### 2.9.6 Bubble World

Bubble World is a query visualization system that allows users to directly interact with the data in a collection of text documents. Chris van Berendonck developed a query visualization tool called Bubble World at the Air Force Institute of Technology as an extension of the InfoCrystal technique. Bubble World makes many improvements on the InfoCrystal technique and provides a robust query tool that queries a repository of textual documents taken from the Los Angeles Times [18].

The knowledge crystallization process, as discussed earlier in this chapter, starts with foraging for data. Bubble World supports this task in a couple of ways. You can first enter a keyword search query to return a list of relevant documents. You can also refine this search through the use of some of the tools.

First, it is necessary to look at how the overview of data is presented. The author makes effective use of many different techniques to discriminate between the points. As seen in Figure 2-7, it is obvious that there are glyphs of different shapes. These shapes are used to denote the number of search words that occur in that grouping. For example, the triangle right in the middle indicates that those four documents contain all three of the search terms. The rectangles on the diagram indicate that those documents only contain two of the search terms. Their placement on the chart tells the user which two words appear. This is a very effective use of glyphs to represent the underlying data.

Another technique used in this representation is the location of the glyph on the graph. This indicates how often each of the words is used compared to the others. If a glyph is closer to one node than another, it contains more occurrences of the first word. These two techniques, glyph shape and location, are used together to give the user an idea of the relevance of that grouping of documents.

These presentation techniques help the user to find relevant information about groups of documents. They also aid the user in searching for schema to compare the documents, which is the second step of knowledge crystallization.



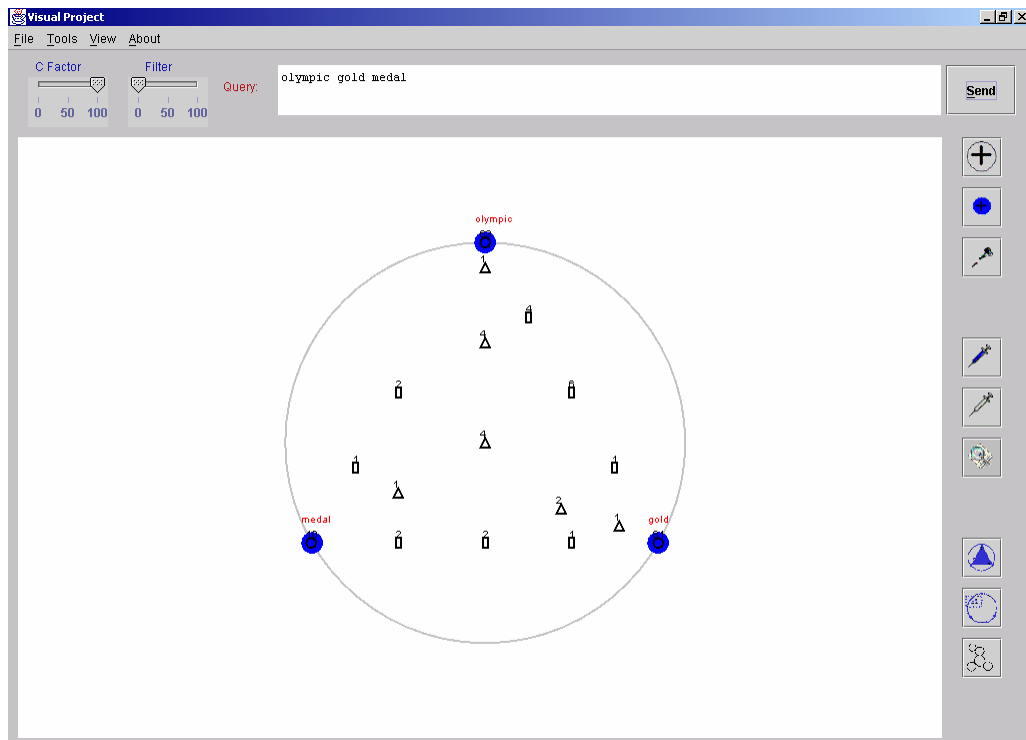


Figure 2-7. Bubble World #1

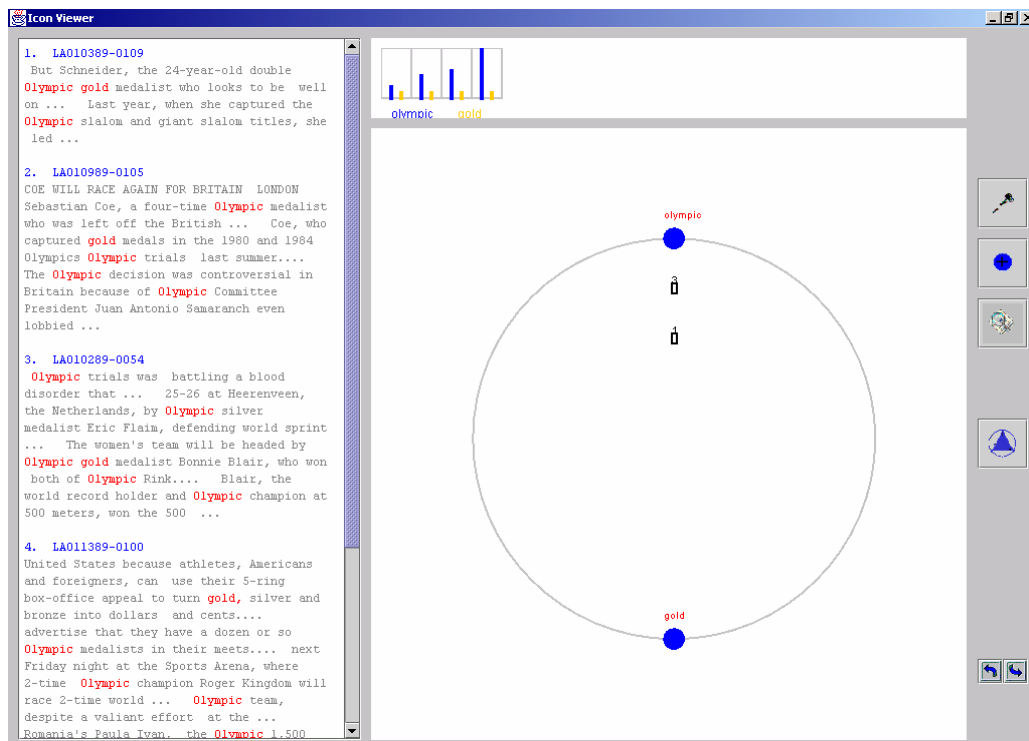


Figure 2-8. Bubble World #2

The third step to knowledge crystallization is to instantiate schema. Bubble World supports this task by allowing the user to drill-down on a specific category. Once the user has narrowed their query down and found the appropriate glyph, they can click on the glyph to get more detail on that group. In Figure 2-8, the detail screen can be seen from one of the groupings in Figure 2-7. Notice, that the new chart shows only the two search terms represented in these documents. On the top of the screen, the user sees bar graphs representing the frequency of each of the search keys in each of the documents. Each graph represents a different document in the group. Each color on the graph represents a search key. This helps the user further narrow down their search. On the left, the user gets a highlighted extract of the document showing some of the context around the search keys. This helps the user determine document relevance as well as find more terms to help in the search. With this information, the user can go back to the first two steps of knowledge crystallization, or continue on with the selected grouping of documents.



**Figure 2-9. Bubble World #3**

In this third step of knowledge crystallization, Bubble World again uses glyphs and locality to aid the user. However, another technique comes in to play here as well. Bubble World uses color to distinguish between entries on the bar graphs. This works fairly well, however the colors chosen might interfere with the user's comprehension of

the data. As seen in Figure 2-9, it is hard to read the titles given to some of the colors. Also, some of the colors are hard to tell apart, like “force” and “marines.”

The fourth step of knowledge crystallization is to problem solve. To aid in this, Bubble World allows the user to click on one of the bar graphs to view a more detailed breakout of the document. It shows where each search key is found in the document. See Figure 2-10. In this step, Bubble World again uses color and location to aid in the knowledge crystallization process. Each search key is assigned a color in the top box. The long rectangle in the top box represents the document as a whole. The color bars in that rectangle represent the search keys and their location in the document. In the lower box, the text of the document appears. The search keys are highlighted in red.

Another technique at play here is overview + detail. In the top box, there is an overview of the entire document, with landmarks showing where the search keys appear. There is also a moving box showing where the focus is at the moment. In the lower box, the text that is in focus is highlighted in purple. This allows the user to browse through the document and easily navigate to the relevant portions. This overview + detail technique is very helpful in this phase, since the user is looking to see if the document matches what they are interested in. However, one downfall that is apparent is that the search keys do not follow the color scheme that they do in the top of the display. When looking for one of the search keys, the user can not just look for a certain color.

The final step in knowledge crystallization is to decide. All of the techniques used up to this point assist the user in this step. Through the use of queries, grouping, colors, drill-downs, overview + detail, and other techniques, Bubble World assists the user in finding exactly what they wanted out of the Los Angeles Times.

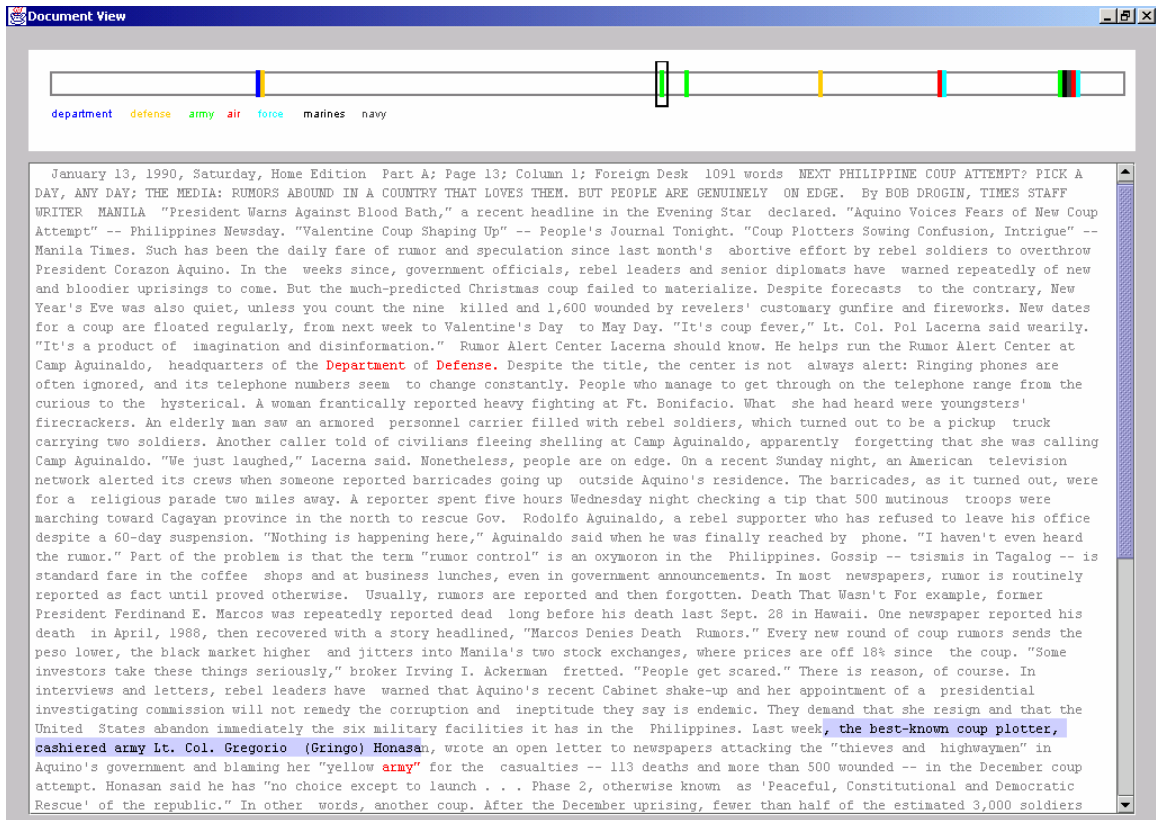


Figure 2-10. Bubble World #4

## 2.9.7 Garlic

Garlic is multi-media query tool, designed to harness the power of multi-media query engines. In today's society, many repositories contain multiple types of data, including images, audio, and other complex documents. Therefore, richer query tools that exploit the intricacies of these data types are being developed. However, a big problem with these queries is integrating them with other data sets.

Garlic is an object-oriented multimedia middleware system that is designed to address this problem. Garlic allows existing data management components, such as a relational DBMS, a full text search engine, or a face recognition system, to be integrated into an extensible information management system. Applications can access any of the data in the underlying data sources through a common, nonprocedural interface, and can exploit the specialized query capabilities of those sources. A single query can access data in several repositories, using the type-specific predicates they support. Garlic also provides a powerful

query/browse application that includes type-specific query interfaces in a uniform query framework [3].

The basic architecture of Garlic is depicted in Figure 2-11. There are several important things to note from this diagram. First, the complex object repository provides linking of several basic documents into complex objects. For example, “a patient's medical folder contains MRI scans (image), lab reports (text), doctors' dictated notes (audio), and address and insurance information (record-oriented database data)” [3]. In this case, the medical folder is a complex object consisting of MRI scans, lab reports, dictated notes, and address and insurance information. This linking allows Garlic to query one type and receive examples of other types. If a user queries on an MRI scan, all medical folders containing matching MRI scans are returned. This added information could help the user query for more relevant data.

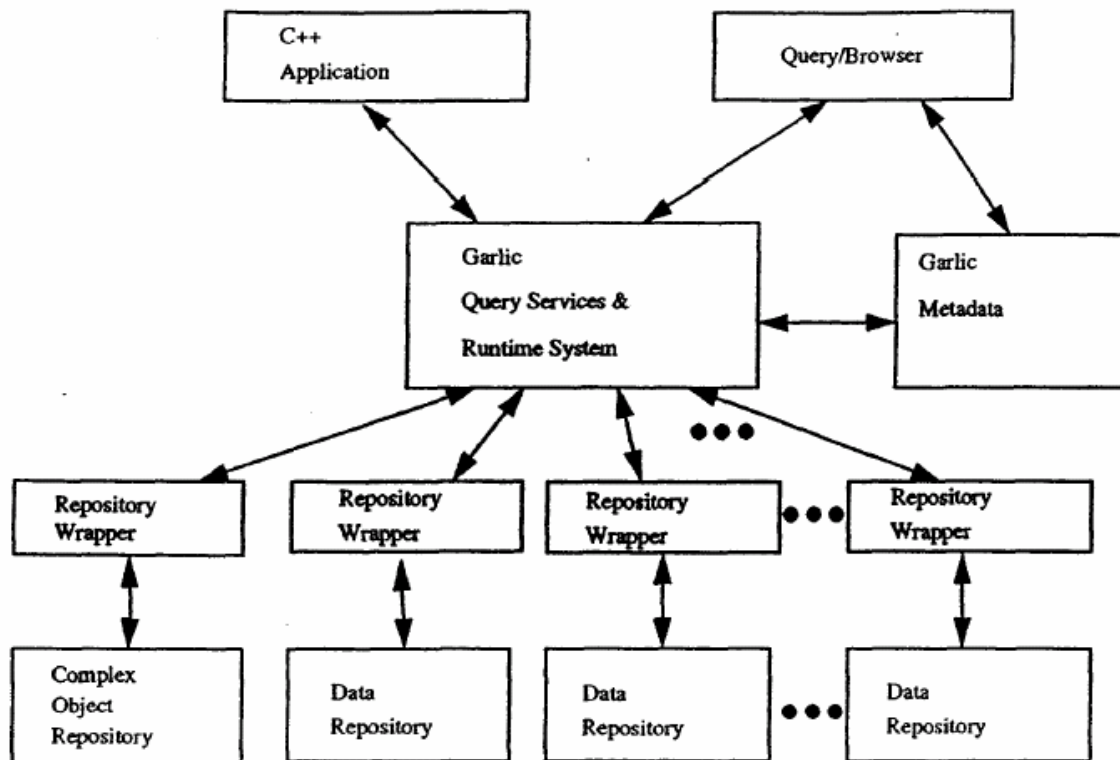


Figure 2-11. Garlic System Architecture [3]

Another interesting thing to note in Figure 2-11, is the storage of metadata. This metadata repository can be used as a transition between types when performing queries. For example, if an image of a sunset were submitted as a query, it is first sent to all image repositories in the system. Once relevant images are retrieved, the metadata associated with those images can be used to query other types. Thus, the returned set may contain images of sunsets as well as poems about sunsets and songs about sunsets.

The strength of Garlic resides in how it wraps each repository rather than each data item. Unlike a system that relies purely on metadata, Garlic maintains the richness of the repository that can be exploited by some of the newer query techniques. Instead of bringing everything down to a common denominator, Garlic attempts to negotiate the richest type of query at run-time. Thus, the results should be more accurate.

## **2.10 Summary**

While there are several techniques and technologies available to help implement this research, no one tool provides all of the functionality necessary to fully implement the JBI concept. Specifically, there are no good tools that provide a visual interface for querying such diverse sets of data. While many good query visualizations do exist, the techniques need to be adapted to the heterogeneous environment of the JBI. This research proposes a technique that does handle this dissimilar environment.

### 3 Design

To help validate this research, it is necessary to develop a software system to implement the ideas and concepts. This system incorporates new and existing functionality to test the areas of research. The system covers a broad base of functionality necessary to fully satisfy the requirements of this research. The functionality is mainly concerned with gathering the required information and processing it. To help define the scope of this system, this chapter lays out the requirements, goals, and design considerations for this system. It also sets the criteria against which this system is evaluated.

#### 3.1 Requirements

There are eight main goals, which were laid out in Chapter 1, for which the system must account. These goals are Information Retrieval, Visualization, User Query Interface, Standard Query Interface, Query Translation, Query Refinement, Resource List Maintenance, and Resource Maintenance. To better define the system for this research, it is necessary to apply the information discussed in Chapter 2 to the goals laid out in Chapter 1. The result is a list of specific requirements for the system to implement.

- Information Retrieval – Based on the constraints laid out in Chapter 1, the system must be able to apply an Information Retrieval (IR) model (possibly several) to the data in the JBI. This model accepts as input a standard query definition language defined for this system and outputs a list of data sources in a standard format. This list is a rank ordered list of matching data with the associated score. Since each field can be of a different data type, the IR model is only concerned

- with querying one field across many objects. Therefore, if multiple fields are queried, each one is handled separately. The results are then combined after each query is completed. This allows multiple data types to be compared to each other.
- Visualization – To satisfy this research goal, it is necessary to provide the user with a visualization that is tailored for the method of Information Retrieval. This means that the system provides an easy to read, multiple-axis display of a large number of data points. The number of axes depends on the number of fields that the user queries. This visualization aids the user in refining the query and picking the most relevant object. Furthermore, the visualization also helps users easily eliminate incorrect query results. To do this, the system implements two functions. First, it provides the user with the means to further clarify a data point on the visualization. Secondly, the system provides a means for the user to modify the query and display the new results.
  - Query User Interface – This interface guides the user through a step-by-step process that determines the information necessary for the query. The required information may vary dependent upon the data type of the field being queried. This interface then compiles all of the pertinent information and poses a well-formed query to the system. The results of this query are then forwarded to the Visualization for display.
  - Standard Query Interface – While developing the Information Retrieval portion of this system, it is necessary to ensure that the IR functionality can be accessed directly. Due to the dynamic nature of the queries and the lack of constraint on types, the IR functionality conforms to a well-defined interface. This allows for



multiple instantiations (i.e. text, numerical, date, etc.) while providing necessary generalization. This standard query interface will be used by future systems to pose queries directly to the underlying layer when no user intervention is required. To conform to system constraints, the results of the query are returned as a ranked list with normalized (0.0 to 1.0 inclusive) values associated with each item.

- Query Translation – The system provides query translators that translate queries of other standards (i.e. SQL, plain text, etc.) into the standard format used by this system. These query translators need to be provided for the integration of legacy systems. Since the two standards may not map directly to each other, this translation might be very coarse and may not provide an ideal solution. The accuracy is based on two main issues. The first of these issues is the degree to which the legacy query standard maps to this system's standard. Secondly, it depends on the implementation used to perform the translation. To conform to system constraints, these query translators are generalized by a standard interface. This allows new query translators to be added at a later date without modification to the system.
- Query Refinement – Query Refinement is necessary due to the dynamic and diverse nature of the data being represented in the JBI. As discussed in the visualization requirement, the user needs a way to refine the query once the results are displayed. This refinement aids the user in more completely defining the query. This produces better results. Possible techniques to be used in this query refinement include using a thesaurus, word discovery, and weighting.

- Resource List Management – This requirement is necessary to support the previous requirements of this system. Without a single list of resources, the other functionality cannot be accomplished efficiently. This resource listing is the data set that is queried by the IR models. The listing maintains metadata on each data point as well as a method of contacting that resource. The listing provides location independence and allows for easy extension. To conform to system constraints, the listing does not limit the data types it allows.
- Resource Management – To satisfy this requirement, the system provides an easy-to-use Graphical User Interface that supports the addition and deletion of resources from the JBI. Specifically, this tool needs to help the user select the metadata to be used for his data. If the metadata template is already available, the publication tool gets this template and allows the user to fill it in. If the template is not available, the publication tool creates a new template based on the user's input and then publishes that template to the system in addition to the publication of the user's data.

These requirements clarify the goals of this research. However, to fully implement these requirements, there are some further considerations that need to be considered. These considerations concern the underlying infrastructure as well as certain methodology decisions.

## **3.2 Design Considerations**

### **3.2.1 General**

During production of this system it is necessary to keep several fundamental considerations in mind. These considerations provide general guidance and help ensure

maintainability and extensibility. The first consideration while developing this system is that each module be developed and packaged separately to aid in flexibility. Each of the previous requirements are satisfied in self-contained entities so that these entities may be used separately or connected in several different configurations. Also, this aids in future extensibility by removing many of the unseen dependencies. Each of these individual pieces contains all error checking and recovery techniques necessary. Each piece is also able to work in an environment of location independence. This means that the information provided by each piece must follow a standard and be unique. Location of resources and services are handled dynamically via the system's look-up service.

To the extent possible, all pieces of this system are programmed to a generic interface to hide implementation details. This goal aids in future upgrades to the system. If the underlying technology is changed, it should be fairly transparent to the users of the system

To implement this system so that it is compatible across multiple platforms and many different configurations, it is necessary to determine a suitable middleware solution. This middleware solution supports location independence, communication between objects, and provides some support for locating necessary resources.

### **3.2.2 Middleware**

As discussed in Chapter 2, there are several possible technologies that provide middleware support for distributed applications. In order to capitalize on existing software, several of these middleware solutions were evaluated to determine suitability for use in this system.

The first middleware solution that was considered was the AFRL JBI JINI-Based Publication & Subscription Services (RJBI). This software solution was analyzed and found to be lacking in several key areas. The most important service that is lacking is messaging. There is no support for communications between resources. Also, the support for notification of resource availability is very inflexible. The only way to receive rapidly changing data (which is a distinct possibility in the JBI) is to continually poll the system for updates. When first registering for a resource, a client can use a listener to wait on the data. However, no such listener exists that notifies a client when a resource updates its data. Finally, the lookup query is not sufficient for application in the JBI. The query only allows exact boolean matching. This does not allow for ranges, searching for a single word in a text field, or partial matches. To successfully implement a query tool, all of these capabilities are needed. Due to these limitations, this solution was ruled out.

Jini<sup>TM</sup>, a Sun Java product, was also evaluated for use as the middleware layer. However, all of the same limitations were found with Jini<sup>TM</sup> as with the AFRL JBI JINI-Based Publication & Subscription Services. It too was ruled out because of these limitations.

Finally, CoABS, the Common Agent-Based System, was evaluated for use as the middleware solution. CoABS, although built on Jini<sup>TM</sup> Technology, does provide the services necessary to implement this system. It contains robust messaging support and allows for change listeners. It does have a slightly better query mechanism, however, it is still lacking. It also has many additional features such as communications protocols and support for agent-based systems that neither of the other solutions had. Most of the

drawbacks of the other two solutions are absent in CoABS, which makes it a good choice. Therefore, the middleware that is used with this system is CoABS.

### **3.2.3 Data**

Another major consideration for this system is the data that is associated with it. This data may take many different forms, such as images, text, audio, simulation objects, etc. With all of these different types of data, some common ground needs to be established. To accomplish this task, a standard metadata scheme is followed for objects participating in the system. The metadata standard to be used needs to provide as much commonality as possible to all objects, without limiting the types of objects that can be added. One way to accommodate such needs is to maintain some very basic information at a top-level and use inheritance to define the rest of the metadata.

## **3.3 Fulfilling Requirements**

Now that guidelines for design have been established, it is necessary to determine the best way to fulfill the requirements discussed in Section 3.1.

The first requirement, Information Retrieval, is heavily dependent upon the structure of the metadata. For this system there are two main dimensions with which the Information Retrieval model is concerned. The first dimension is that of querying a single field across multiple metadata objects. Of the two dimensions, this one is the most aligned with conventional Information Retrieval techniques. Since the fields are common across the objects, each of the fields being queried is of the same data type. However, this does not mean that all fields must be of the same type. For example, assume the system is querying several metadata objects that are instances of a *PERSON* object. A single dimension of this query is to query all of the objects based on the *NAME* field.

Since all of the *NAME* fields are Strings, they can all be queried as text. However, this does not mean that the *BIRTHDATE* field must also be text. The only requirement is that each query returns a score for each item in the range of 0 to 1 inclusive.

Once each of the queries is run on the individual metadata fields, the results can be combined. Since each query returns a normalized value, the queries can now be treated equally. This comprises the second dimension of this system's Information Retrieval process. Once the results are calculated for each individual field, the results need to be combined into a meaningful and coherent format to aid the user in selecting the most relevant object.

To obtain the information from the user necessary to perform the Information Retrieval process, the system employs a wizard concept that guides the user through the information gathering process. This wizard needs to adhere to some basic visualization guidelines in order to be effective. In their paper, "Improving a Human-Computer Dialogue" [20], Molich and Nielsen propose nine heuristics for evaluating the usability of user interfaces. These nine heuristics are used to evaluate this system.

- Simple and Natural Dialog – The authors state that extraneous information in dialogs actually detracts from user comprehension and usability. It is best to keep the information and instructions very simple for the user. Also, the authors point out that all information should be in a natural and logical flow. This aids in quick and accurate responses by the user.
- Speak the User's Language – The authors show that information should be laid out in terms and ways that the users of the system can understand. Often

developers put “system-oriented” [20] terms in the dialogs that the users do not understand.

- Minimize the User’s Memory Load – This heuristic seeks to minimize the amount of information a user must remember from one step to another. This includes instructions given at the beginning of several steps. If the instructions are complex, the authors suggest simplifying them and displaying them as needed.
- Be Consistent – According to the authors, consistency has two main scopes. First, within the system, all terms, commands, and actions should be standardized throughout the system. This keeps a user from guessing the correct action to be taken. The second scope incorporates other commonly used systems and sub-systems. If there are standards in the domain that the system is concerned with, they should be adhered to, allowing users to quickly learn the system.
- Provide Feedback – In order to keep the user informed, the system should provide meaningful and timely feedback when the user completes an operation.
- Provide Clearly Marked Exits – Since users often “choose system functions by mistake” [20], it is necessary to provide an easy way for the user to get back to a familiar state. The authors go on to say that this process should not require the user to endure a lengthy process while doing this.
- Provide Shortcuts – This heuristic allows both novice and expert users to work easily with the system. By providing shortcuts, the expert users can by-pass steps

they do not need to fulfill their task. It provides a quicker way for the expert to get the job done, while not hampering the efforts of the novice user.

- Provide Good Error Messages – Good error messages aid the user in learning the system and determining what went wrong. “Good error messages are defensive, precise, and constructive. Defensive error messages blame the problem on system deficiencies and never criticize the user. Precise error messages provide the user with exact information about the cause of the problem. Constructive error messages provide meaningful suggestions to the user about what to do next” [20].
- Error Prevention – This heuristic emphasizes the need to aid the user in avoiding errors. This can be accomplished through intuitive and useful design and instructions.

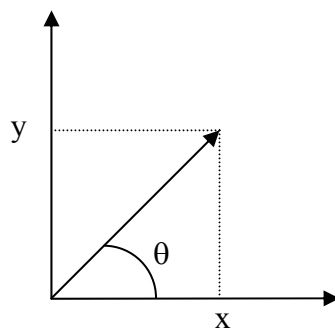
For this system, a visualization is also used to combine the results from the Information Retrieval process and display them to the user. This visualization also needs to adhere to the nine heuristics listed above as well as provide a meaningful layout of the query results. In Chapter 2, several existing techniques for visualizing query results were explored. The two most viable options were InfoCrystal and Bubble World. Both of these options plot relevance on multiple axes distributed around a central point. Position around the center point determines the type and degree of matching that each data point achieves. This visualization technique seems to be fairly effective in accurately portraying the relevance of the data.

In evaluating these two tools, it was found that neither provides all necessary aspects for this system. The first system evaluated, InfoCrystal, does not work due to the



limitations placed on query results. The main limitation placed on the queries is the fact that they have to be boolean queries. More precise queries cannot be used with this visualization. The second system, Bubble World, accounts for this problem. Bubble World improves on the concepts of InfoCrystal by accounting for more precise queries. This allows much finer granularity on grouping and plotting the data points, thus providing a more usable output. However, Bubble World has one major drawback in that it only allows word queries on textual documents. This is far too limiting for the types of data that comprise the JBI.

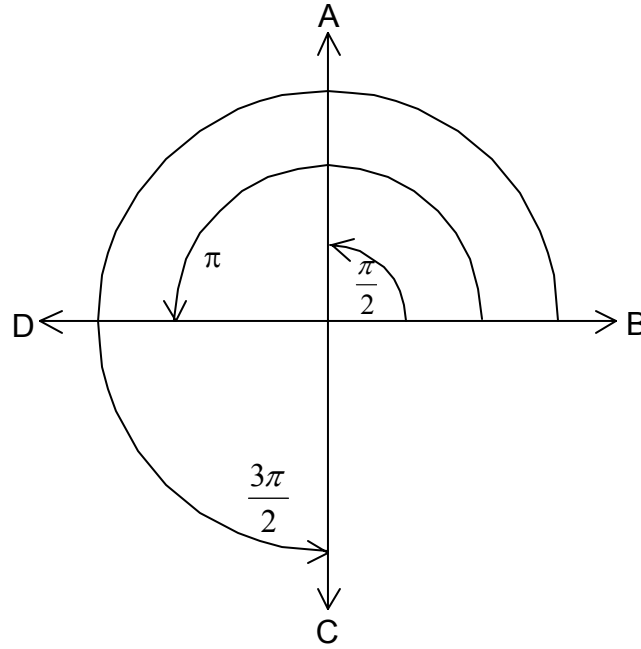
Although neither of these two tools can be used directly, concepts from each can be applied to this system. In this research, a system of evenly spaced axes arranged in a circle is used for the visualization of the queries. Each axis represents one of the query fields. To plot each data point, the system calculates the x- and y-components of each of the vectors created by the relevance scores on each axis as shown in Figure 3-1. These x- and y-components are summed to determine the final position for each data point.



**Figure 3-1. Plotting Example**

As an example, suppose there are four fields of interest: *A*, *B*, *C*, and *D*. Also, suppose that that data point *p* has relevance scores of 0.75, 0.3, 0.15, and 0.8 for *A*, *B*, *C*,

and  $D$  respectively. To plot  $p$  on the visualization, the first step is to find the angles of each axis from the positive x-axis. These angles,  $\theta$ , (in radians) are  $\frac{\pi}{2}$ ,  $0$ ,  $\frac{3\pi}{2}$ , and  $\pi$  respectively as seen in Figure 3-2.



**Figure 3-2. Angle Calculation**

Once the angles are calculated, the next step is to calculate the x and y contribution of each relevance score. This is done using Figure 3-3 where  $r$  is the radius of the plot area and  $z$  is the relevance score with respect to a single field.

$$x = r \cdot z \cdot \cos(\theta)$$

$$y = r \cdot z \cdot \sin(\theta)$$

**Figure 3-3. Contribution Equations**

For the example given above, these equations need to be used four times, one for each field, to calculate where  $p$  should be plotted. For this example,  $r$  is equal to 10. This process is depicted in Figure 3-4.

$$\begin{array}{ll}
 x_A = 10 \cdot (.75) \cdot \cos(\frac{\pi}{2}) = 0 & y_A = 10 \cdot (.75) \cdot \sin(\frac{\pi}{2}) = 7.5 \\
 x_B = 10 \cdot (.3) \cdot \cos(0) = 3 & y_B = 10 \cdot (.3) \cdot \sin(0) = 0 \\
 x_C = 10 \cdot (.15) \cdot \cos(\frac{3\pi}{2}) = 0 & y_C = 10 \cdot (.15) \cdot \sin(\frac{3\pi}{2}) = -1.5 \\
 x_D = 10 \cdot (.8) \cdot \cos(\pi) = -8 & y_D = 10 \cdot (.8) \cdot \sin(\pi) = 0
 \end{array}$$

**Figure 3-4. Example Calculations**

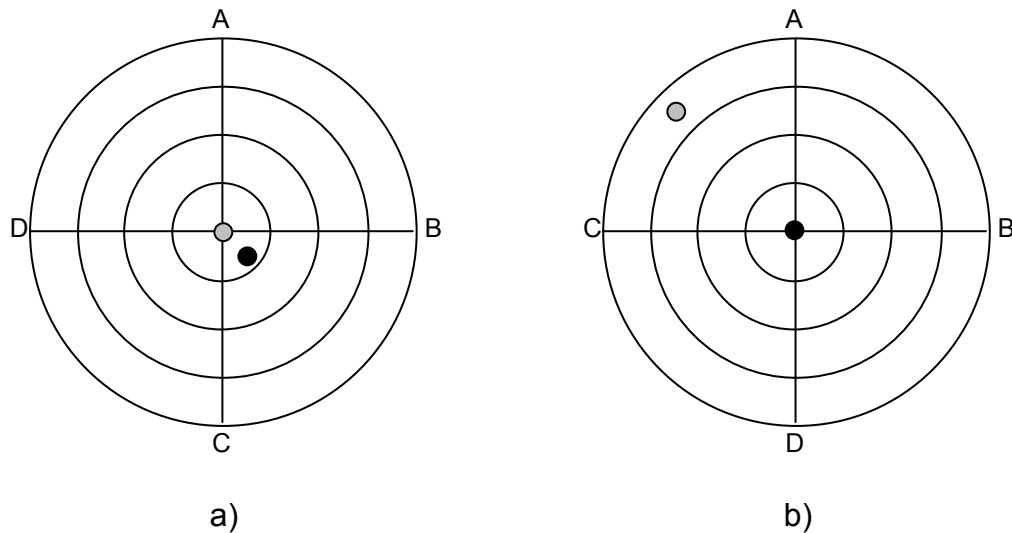
The final step in calculating the x- and y-coordinates of  $p$  is to sum all of the x contributions and sum all of the y contributions. This is depicted in Figure 3-5. Using this method,  $p$  should be plotted at (-5,6). This process is repeated for each data point.

$$\begin{array}{l}
 x_p = x_A + x_B + x_C + x_D = 0 + 3 + 0 + (-8) = -5 \\
 y_p = y_A + y_B + y_C + y_D = 7.5 + 0 + (-1.5) + 0 = 6
 \end{array}$$

**Figure 3-5. Example Summation**

There is a fundamental weakness with this approach, however. Since the concept of the “bull’s eye” chart is to bring relevant points to the center and push less relevant points to the outside, it is expected that the point closest to the center is the most relevant. However, this is not always true. This model leaves room for less relevant data to be plotted close to the center of the circle. This is caused by using a 2-Dimensional (x,y) coordinate axis system to represent the data. Each dimension has two distinct directions of pull. This means that more than one query field can move each data point along a

single axis. If the fields were distributed such that opposing fields have comparable relevance, then the point gets plotted fairly close to the middle. However, this is not always desirable. Figure 3-6 illustrates this point. The only difference in a) and b) is the location of the query fields along the outer circle.



**Figure 3-6. Plotting Error Example**

In this example, the gray dot has a relevance of 1 with respect to A and C and a relevance of 6 with respect to B and D. The black dot has a relevance of 6 with respect to A and D and a relevance of 5.5 with respect to B and C. As you can see, when the fields with similar relevances are opposite each other, the point gets plotted in the center of the diagram. When they are adjacent to each other, the data point gets pushed away from the center. This can produce a very undesirable outcome. A more accurate depiction of the gray data point is (b) and a more accurate depiction of the black data point is (a). However, when comparing gray to black, the black dot has much more relevance than the gray dot. This is not conveyed in (a).

At first glance, this looks like an easy problem to solve: just make sure that opposite relevances are not equal (unless they are all roughly equal). However, this only satisfies the accuracy of one data point. On this graph, there is expected to be anywhere from 10 to 10,000 or more data points. So, it is easy to see that this solution does not work.

The root of the problem is that the most relevant item is expected to be in the center. It is easy to see that the more relevant items will always plot toward the center of the graph. However, less relevant points can also plot towards the center causing confusion. To minimize the problem an aid to the user is necessary.

Since this system is designed to be interactive, the system plots the graph normally. Then, the user is able to select a data point, which activates a feature that displays a polygon of the relevance of that data point with respect to each field. This polygon has vertices along each fields' axis that indicates the relevance to that field. These points are then connected. This allows the user to summarize a few things about each data point. The first thing that can be summarized is based on size. The more area that the polygon fills, the more relevant the associated data point. Secondly, the more perfectly shaped a polygon is, the more evenly its relevance is distributed among the fields. Adding this technique to the visualization allows the user to quickly rule out any "distracters" that may erroneously plot towards the center of the graph.

In addition to a user interacting with the system through the Query Wizard and Results Visualization, an automated means of accessing the system must also be provided. This ensures that future programs designed to integrate with this system can easily and efficiently access the information they require. Since the Query Wizard needs

an interface to submit queries, a standard programming interface already exists and it is only necessary to evaluate whether this interface meets this requirement or not. Based on the requirement as laid out previously in this chapter, the interface must provide a way to execute a query and return a ranked list with associated normalized values for each object. Since the Information Retrieval is different for each different data type, a standard programming interface is established that can be implemented for each different data type. This satisfies the requirement for a standard programming interface.

For the requirement of query translators, it is also necessary to define an interface that translates a query to the standard language and executes it. In some cases it is necessary to translate the single query into multiple queries and combine the results. All of this can be done with an interface that provides a generic query method. An important thing to point out is that the input parameter is kept as generic as possible to allow maximum flexibility when executing the query. Each different translator may require different parameters, but all can still be generalized. The result of this method is a mapping of terms to values. This allows for queries to be performed on a single field or multiple fields. The values assigned to each term are normalized (0.0 to 1.0 inclusive). The method of determining these scores will vary greatly between different implementations.

The next requirement to be satisfied is that of query refinement. This requirement takes on many forms for this system. The first and most rudimentary form is incorporated into the Query Visualization. This includes being able to select which types are displayed as well as the polygonal shading of each data point to depict its level of matching for each query field. The more involved form of query refinement re-guides

the user back through the information gathering process, providing suggestions of what to change to improve the query results. This process follows the same structure and guidelines as the information gathering GUI. The main difference is that there are some Information Retrieval techniques incorporated to aid the user.

One very important requirement of this system is that of providing a service that maintains a list of all of the objects available in the system. This service aids in location independence as well as handles dynamic data changes. The lookup service included with CoABS provides all of the necessary functionality to meet this requirement. The service maintains a searchable listing of all resources in the system along with associated metadata. By using this lookup service, more efficient communications can be achieved.

The final requirement of the system is that it provides a service in which to publish data and from which to subscribe to data. Again, CoABS already has this built in to it. When analyzed against the requirements, the CoABS implementation is definitely suitable for use in this system.

### **3.4 Conclusion**

The design discussed in this chapter provides the initial framework from which this system is developed. The key to this design is that it is based on interfaces. This generic approach to system design allows for future expansion and modification with very little effort. It also provides the flexibility necessary to represent the virtually limitless type of data that may be introduced into the JBI.

## 4 Implementation

### 4.1 Existing Technologies

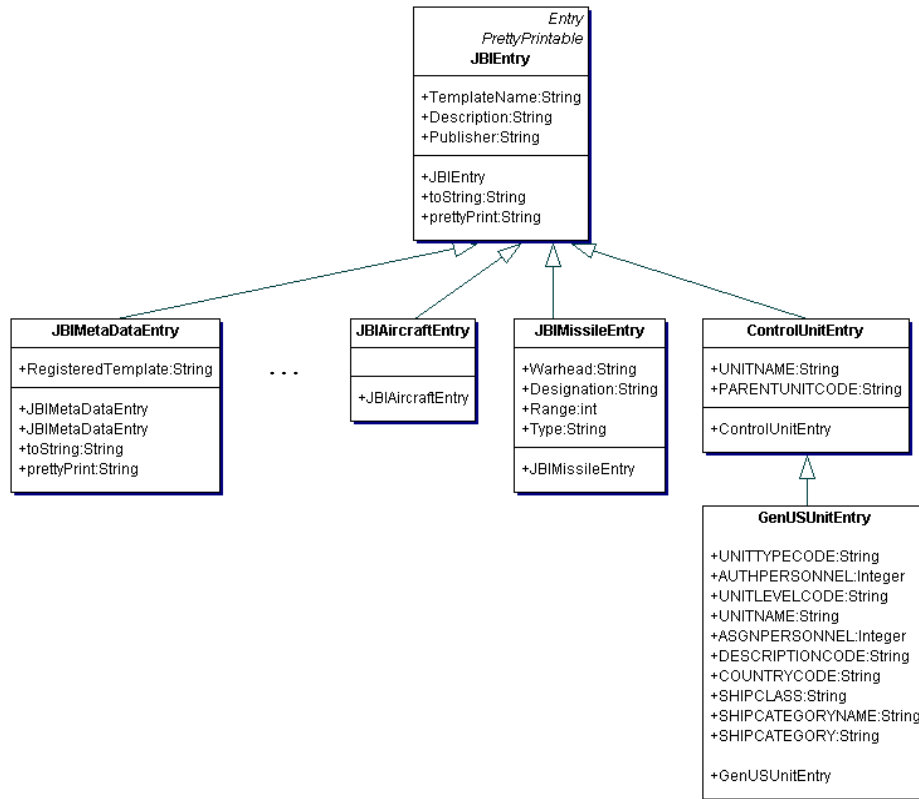
As discussed in Chapter 3, the only existing technology that is used in this system is the CoABS Grid. This grid provides the middleware necessary to abstract the location and communications issues inherent in a distributed system, such as the JBI. The main functionality used by this system includes registration, resource locating, and messaging. Several of these functions actually make references to the underlying Jini<sup>TM</sup> framework. However, even these calls are considered part of CoABS, since CoABS should eventually provide this implementation.

### 4.2 Metadata

In order to implement the functionality of this system, it is first necessary to implement the underlying metadata structure. This structure dictates how the rest of the system is implemented. Therefore, for this system, the metadata structure in Figure 4-1 is followed. This structure was derived from the Dublin Core Metadata Initiative (DCMI). The DCMI bases its metadata schema on a core set of identifiers that completely describe all resources. One of the premises of DCMI is that the smallest amount of metadata that completely describes the necessary data should be used. This provides maximum flexibility with minimal overhead.

DCMI proposes 15 core metadata tags that should be used as the base set of identifiers. For this research, however, a much smaller set of core identifiers is used. These identifiers are maintained in the *JBIEntry* object. Another premise of the DCMI is that these identifiers are only the core set of identifiers. DCMI recognizes the need to





**Figure 4-1. Metadata Structure**

have additional (possibly dissimilar) identifiers. These identifiers aid systems in better describing their data. Following this concept, this research implements inheritance off of the root object to add more identifiers. This allows for better classification and description of services, as seen in Figure 4-1.

In this scheme, *JBIEntry* and *JBIMetaDataEntry* are standard classes that accompany this system. The other entries serve as examples to depict how this scheme can be used for expansion and are not part of the final system. By using this scheme, a user can specify which metadata object he wants to query with, or use the standard *JBIEntry* fields to execute a query. This metadata standard is the foundation for this

entire system. Each object participating in the system needs to have a proper metadata object associated with it.

### **4.3 Information Retrieval and Standard Interface**

With this common metadata baseline, it is possible to implement the Information Retrieval requirement. Although laid out as two separate requirements, the Information Retrieval requirement and the Standard Query Interface requirement are best dealt with at the same time. They each affect the implementation of the other.

Keeping in mind the goal of not limiting the future potential of this system, it is necessary to define a standard interface for each query type that can be implemented in as many ways as is necessary. To support this, two things are necessary. First of all, it is necessary to create a class that all of the query classes can inherit from. This allows all queries to be treated equally, while at the same time providing the independence to implement each query separately. Figure 4-2 shows the design for this standard interface. To build a query, the system creates a new *JBIField* of the correct type (based on the data type of the field being queried). Next the system fills in the appropriate values (which can be done by the user in the provided GUI). Next the system executes the query by passing a list of objects to query. The result is a mapping of objects to relevance scores.

The second thing that is necessary to implement this query system in a generic nature is a factory method that provides the appropriate *JBIField* when requested. This reduces the amount of work necessary to add new data types to the pool of available types to query. This factory can be incorporated with the top-level query interface. The design for the standard query interface is as shown in Figure 4-3.

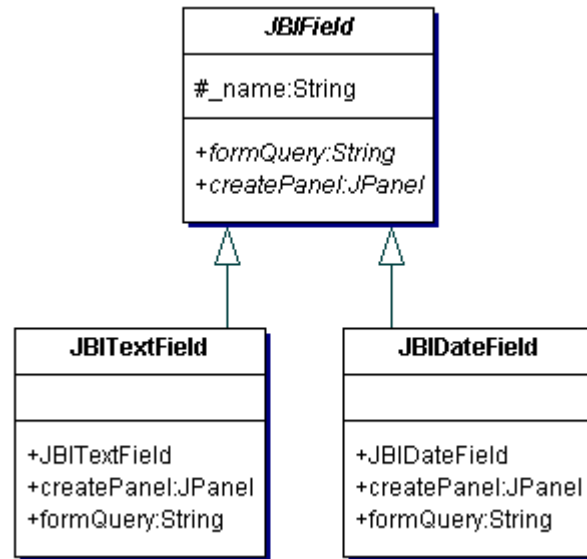


Figure 4-2. Structure for Standard Query Interface

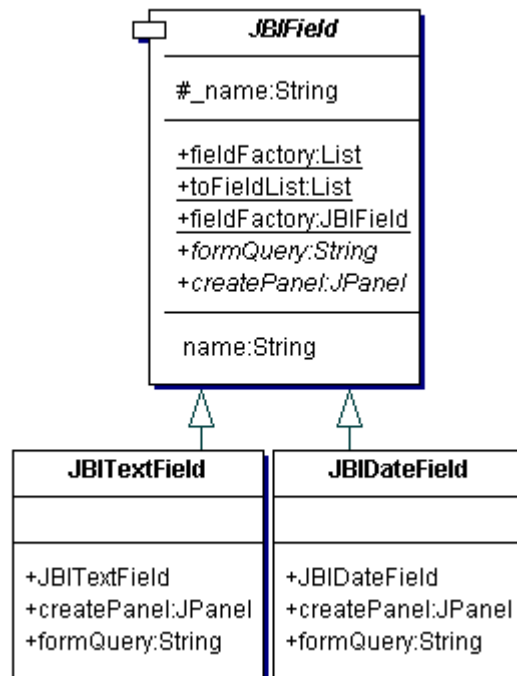


Figure 4-3. Updated Structure for Standard Query Interface

#### 4.4 Query User Interface

To aid the user in supplying useful query information, the system contains a Query Wizard that steps the user through a process to get the required information. To better illustrate the functionality of this system, the following scenario serves as an example of the functionality. A user wishes to find a missile object in the JBI. The missile needs to carry a conventional warhead that is armor-piercing. Furthermore, the missiles range must be between 300 and 500 miles and able to be launched from the air or ground. The following paragraphs illustrate the steps the user must take to satisfy this scenario.

The first step of this wizard, as seen in Figure 4-4, helps the user determine the type of query he would like to perform. As seen, there are three options: Generic, Combination, and Specific. The Generic query allows users to search all of the resources in the JBI using only the basic JBI metadata. The Specific query method allows users to narrow the search to a specific data type. This allows the user to use the more specific metadata tags that are found within that data type only. The Combination query method allows users to search all data types, just as Generic does. However, it allows for more detailed searches for a data type that the user specifies. If Generic query method is chosen, the user is automatically taken to step #3 (Figure 4-6). Otherwise, the user continues to step #2 (Figure 4-5). For this example, the user selects the Specific Query option since he knows he is looking for a particular type of object (missile) and presses the 'Next' button.

In the second step of this wizard, the user is asked to select a metadata type from a list of current types published in the JBI. This list is compiled at run-time by requesting

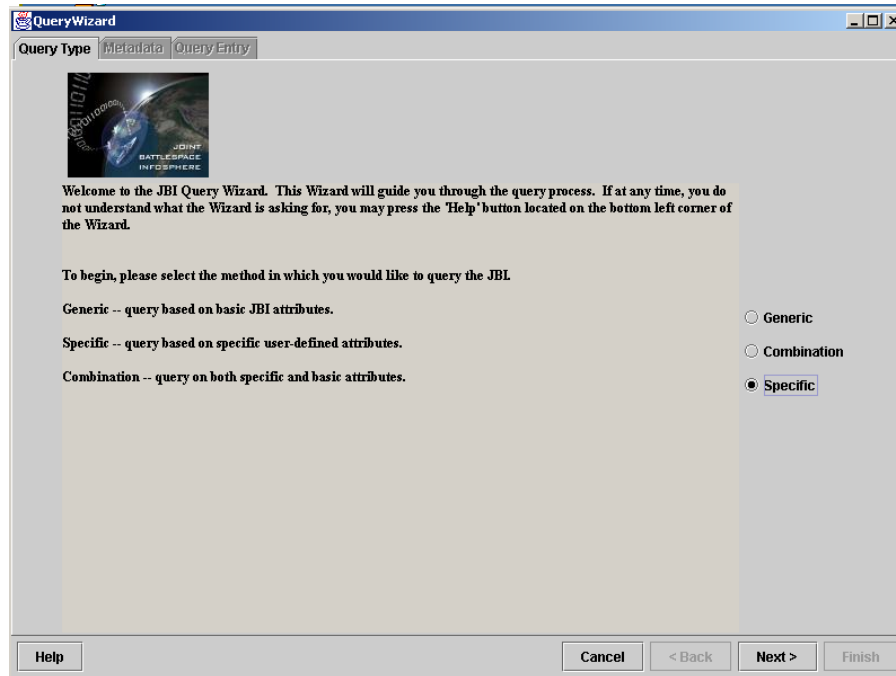
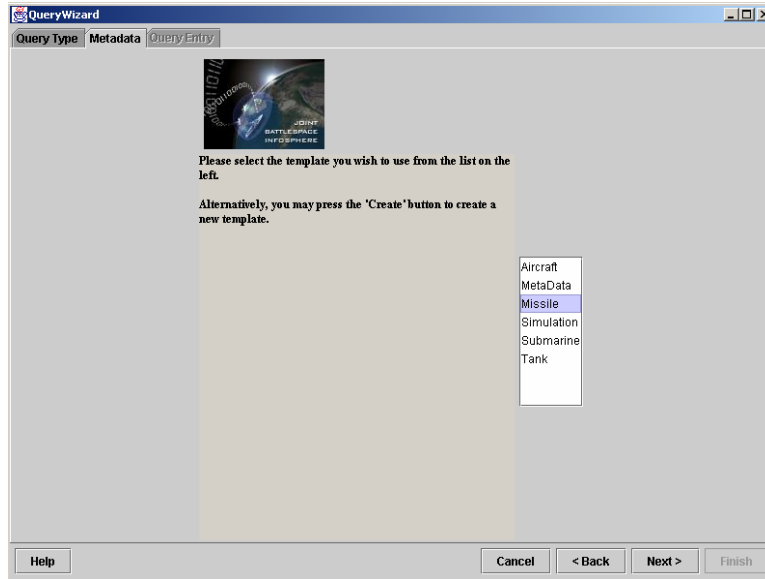


Figure 4-4. Query Wizard Screen #1

all metadata entries from the JBI. Since the user knows he is looking for a missile, he selects the Missile entry and presses the ‘Next’ button. Once the user selects the template he wishes to use, the wizard proceeds to step #3 (Figure 4-6).

In step #3 (Figure 4-6), the user is presented with all of the fields that are appropriate based on the decisions made in step #1 and step #2. These fields are generated on the fly as they are added to the screen. This is done using the *JBIField* structure to allow maximum flexibility for new data types. As can be seen in Figure 4-6, the *Warhead* field is different from the *Range* field. The first is a text field and the latter is a numerical field. By generating these fields on the fly, the system is able to handle any data type that has been introduced into the *JBIField* structure (Figure 4-2). The type of *JBIField* is determined by a factory method contained within *JBIField*. The user



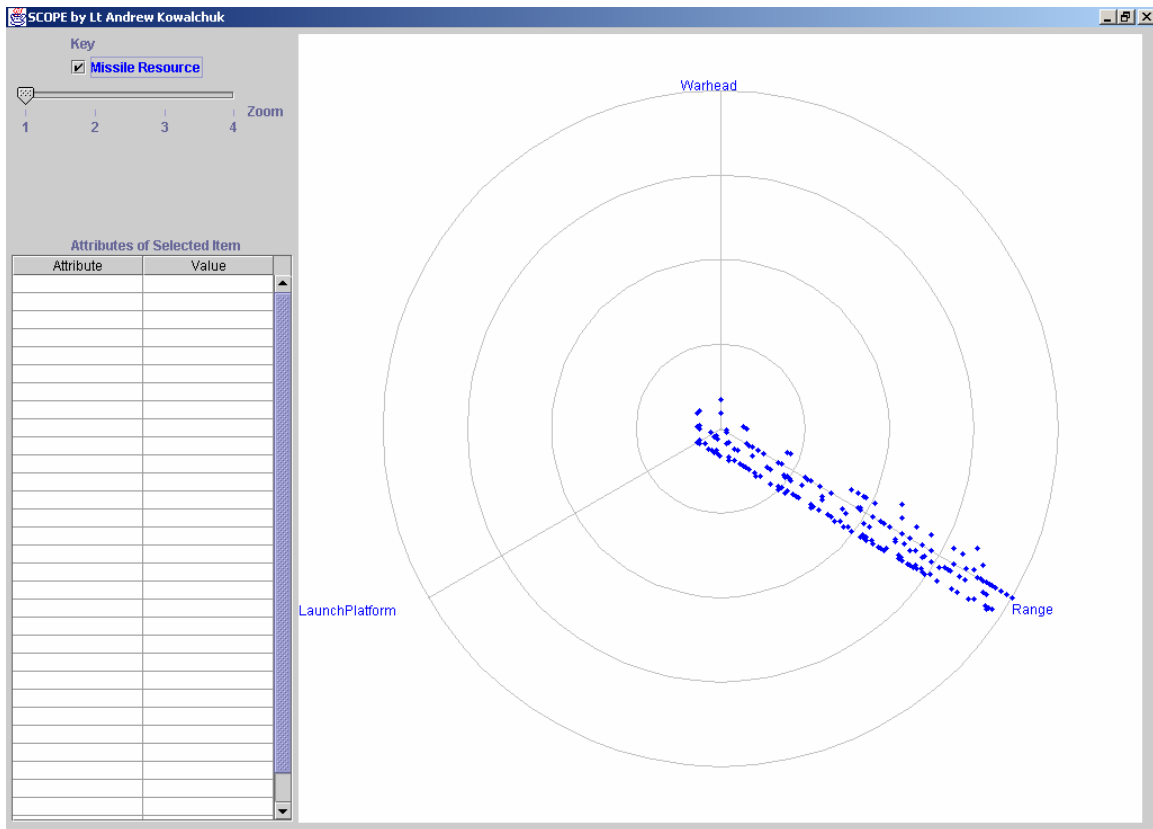
**Figure 4-5. Query Wizard Screen #2**

enters the information he is looking for in the appropriate fields. Blank fields are not searched, since the user did not enter any information. Once the user enters the information, he presses the 'Finish' button. This takes him to the visualization of the query results.

**Figure 4-6. Query Wizard Screen #3**

## 4.5 Visualization

As discussed in Chapter 3, this system uses a circle segmented into equal portions to display the results of the queries, as seen in Figure 4-7. In this circle, each axis represents a field that the user queried. If a field is not present, the user entered no query information about that field. Displaying these unused fields severely detracts from the readability and usability of this display.



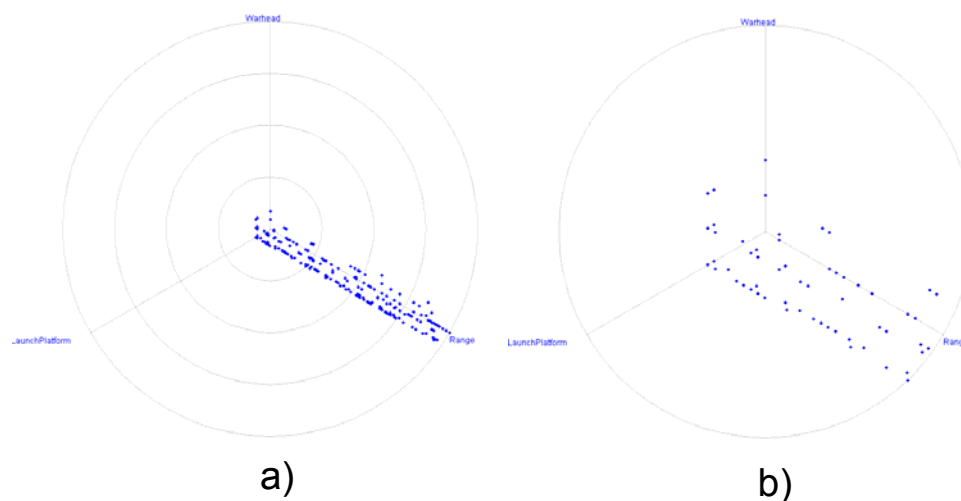
**Figure 4-7. Visualization Screenshot**

The main part of this display is located on the right half of the screen. It is composed of several concentric circles cut by evenly spaced axes. The number of axes depends on the number of fields that were queried using the Query Wizard. Each data

point is plotted individually based on its relevance to each of the queries (represented by the axes). The method used to plot the points is discussed in Section 3.3 of this paper.

There are several tools provided to the user to aid in the selection of the proper data point. The first tool can be found in the upper left-hand corner (Figure 4-7). This series of check boxes allows the user to turn data points on or off. In this example, only one data type is displayed, so the only check box available is that of “Missile Resource.” If the box is checked, all of the matching data points that are of that type are displayed. If the box is not checked, the visualization does not display any results of that type. This legend also depicts the color-coding used for different data types.

Right below the legend is a zoom bar. This bar allows the user to change the scale of the display on the right. As the zoom is increased, data points that no longer fall within the displayed range are no longer plotted. This helps eliminate the clutter on the screen. As seen in Figure 4-8a, the data points that are in the inner ring are very cluttered. It is hard to distinguish between the different points. In Figure 4-8b, the display is set to full zoom. Now it is very easy to pick out individual data points.



**Figure 4-8. Example of Zoom Capability**



Now that this information is presented to the user, he can now go through the data points and find the one he wants. To do this, he only has to click on one of the data points on the display. Once the user clicks on a data point, two things are displayed about that data point (see Figure 4-9). First, the metadata associated with the resource represented by the data point is displayed in the table in the bottom left-hand portion of the screen. This allows the user to “browse” through the results to find a suitable resource. Also, a shaded polygon is drawn on the display that shows the relevance scores for each axis. This aids the user in eliminating “false-positives” that may be displayed. In this example, the user has found a land-based missile object with a range of 490 miles that carries armor-piercing warheads.

### Figure 4-9. Detail of Single Data Point

## 4.6 Query Translator

Due to the nature of legacy system integration, it is impossible to know all of the different kinds of queries to be translated. Therefore, this system provides an interface that allows new translators to be added to the system very easily. All of this can be done within the interface depicted in Figure 4-10. An important thing to point out is the use of *Object* as the input parameter when executing the query. Also, the output is a *Map* that associates fields with relevance scores. This generic structure provides the necessary functionality without creating any unforeseen dependencies.

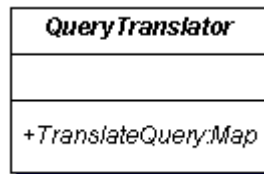


Figure 4-10. Query Translator Structure

## 4.7 Query Refinement

This system does not implement any form of intelligent query refinement in its current state. However, it does allow the user to revisit his query input and make modifications. At that point, the query is recalculated and then displayed to the screen. Suggestions for future modifications of this aspect are contained in Section 5.3.5 of this paper.

## 4.8 Resource List Management

This requirement is fulfilled by CoABS using the Jini<sup>TM</sup> Lookup Service (LUS). The service provided by the LUS does everything necessary to fulfill this requirement. The LUS tracks resources in the system and maintains a proxy that can be used to communicate with the resource. When a resource is requested from the LUS, the proxy

is then passed to the requestor. This allows direct communication between the requestor and the resource, cutting down on the demand on the LUS.

#### **4.9 Resource Management**

CoABS also fulfill one aspect of the resource management requirement. CoABS uses the Jini<sup>TM</sup> LUS and Java RMI services to manage the active resources. The main mechanism used to keep the list accurate is leasing. When a resource registers with the system, it is given a lease on its entry in the LUS. If it is still active, it can keep renewing this lease, thus staying active in the system. However, if the resource fails to reregister or terminates prematurely, the lease expires and the resource is removed from the list. This ensures that the list is not stale.

The other aspect of this requirement is that the system provides a Graphical User Interface (GUI) to add resources to the system. To satisfy this requirement, the system contains a Publication Wizard. Like the Query Wizard, this Publication Wizard steps the user through the process of adding resources to the JBI. The first step of this wizard, shown in Figure 4-11, prompts the user to select the metadata template to be associated with his resource. If the template he needs is not available, he can create a new one using the button on the wizard.

If the user found the template he wanted, he continues to step #2 of the process, depicted in Figure 4-12. In this step, the user fills in all of the information concerning his resource. This information then conforms to a standard metadata template for that type of resource. Additionally, these templates are acquired in real-time. Therefore, new additions to the system are handled with ease.

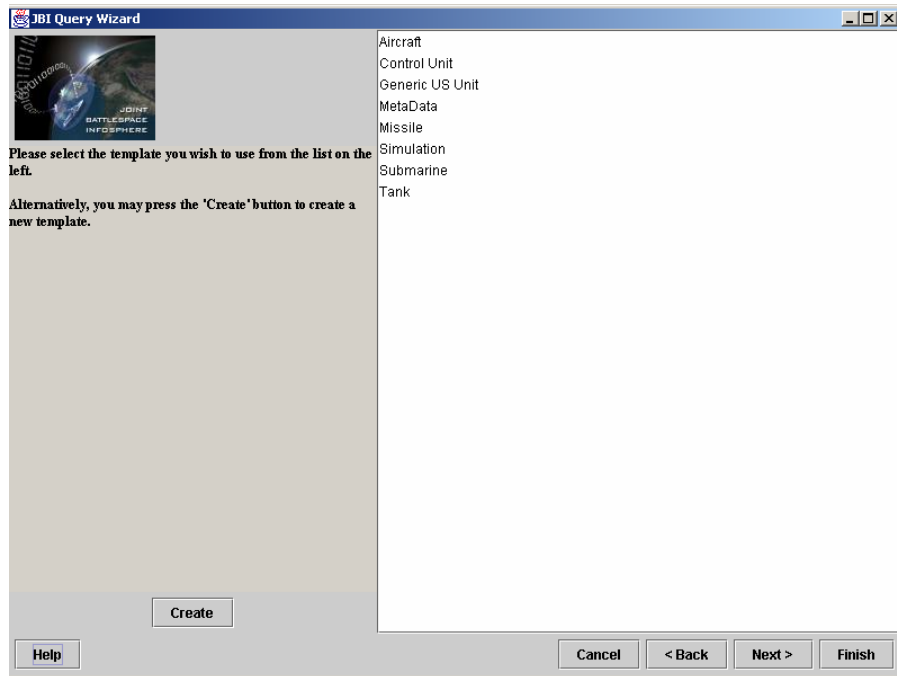
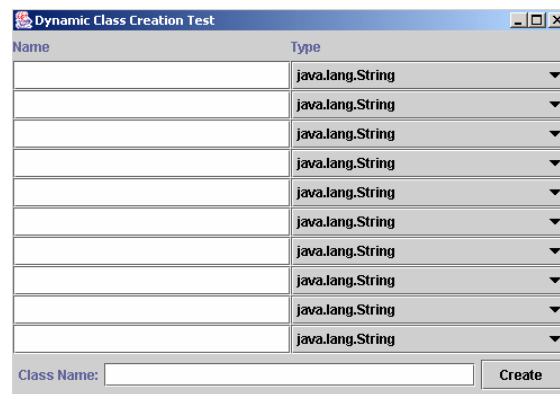


Figure 4-11. Publication Wizard Step #1

Figure 4-12. Publication Wizard Step #2

If the user does not find the metadata template he needs, a metadata template creation tool is provided to aid the user in creating and populating a metadata template. This tool, shown in Figure 4-13, provides a means to let the user specify the attributes associated with the template, as well as a name for the template. This tool then creates a Java Class file containing this information and publishes the template to the JBI. Therefore, once one user has created a template, it is available to all users. This helps ensure seamless integration of the many systems in the JBI.



**Figure 4-13. Metadata Template Creation Tool**

## 4.10 Summary

This system, as detailed in this chapter, provides the initial framework for implementing queries on a large heterogeneous data set such as the JBI. It takes into account the dynamic nature of such a large distributed system, yet provides an easy to use interface for the user. This implementation provides both power and flexibility to the query mechanism. This mechanism goes beyond most metadata query systems. It allows for multiple types within the metadata. The only constraint is that a query function must be written for each type. However, the framework supports this through generic interfaces.

### **5.1 System Evaluation**

To evaluate the effectiveness of this system and determine the value of the research, it is necessary to compare the completed system to the original goals of this research, as laid out in Chapter 1. This evaluation proves that the concept of this research is feasible and highlights the gains made by this research. To accomplish this evaluation, the system is examined against all of the requirements to determine how well it meets the requirement.

#### **5.1.1 Information Retrieval**

The need for Information Retrieval (IR) is prompted by the fact that the JBI concept is designed to house vast amounts of resources. Many of these resources rely on other resources to function. Thus, they need to be able to find each other. While IR itself is very well advanced, one key area that still remains relatively uncharted is the domain of heterogeneous data. This domain presents many problems that prevent standard IR techniques from being exploited. Currently there are no tools available that provide search capabilities for unlimited data types without using metadata.

Similarly to those tools, this research employs a technique that uses metadata to “equalize” all data types. This does detract somewhat from the richness of the data set, however it allows comparison between unlimited data types. Unlike many of the tools available, this research does not treat the metadata as text only. Using Java Objects and inheritance, this research allows multiple data types to be used as long as a query strategy exists for that type.

Another unique contribution of this research is the way queries are executed. In order to exploit the use of multiple types in the metadata, each field of the metadata is treated as a separate query. This provides a much richer query environment and makes the results easier to work with. By treating each field separately, it is possible to exploit more specific query methods for each data type. For example, in this paper, an example was used that showed a query with textual fields as well as numerical. In the textual fields, the system implemented a vector model for searching the field. On the other hand, the numerical fields were searched using a range and proximity to the center point. Since each field is its own query, results can be displayed using mainstream multi-dimensional query visualization techniques. This provides the user with a more complete and accurate picture of the data.

#### **5.1.2 Visualization and Query User Interface**

This research has focused heavily on the use of visualizations to gather query information and display the results in a usable manner. By using this graphical output, the user is better able to comprehend the results and make a more informed decision. To evaluate the system against these requirements, it is necessary to use the heuristics described in Section 3.3.

- **Simple and Natural Dialog** – Throughout this system, the dialog boxes are kept very simple and do not use complex terminology to convey the message. In addition, the structure of each dialog is segmented in a way that makes it easy to follow the instructions. Special emphasis was placed on minimizing the amount of dialog needed to complete the tasks. To aid in this, simple instructions are presented throughout the wizard. More detailed help is contained in special help

screens. This keeps the basic information from being cluttered, while also providing the detail that may be necessary. This system follows this heuristic very well.

- **Speak the User's Language** – Similarly to the previous heuristic, this system also follows this one very well. Only terms that are familiar to the most novice user are used in the system. The one aspect of the system that might not follow this heuristic is the field names. Since users define these names, they may or may not be self-explanatory. For example, a user might name a field *lgtevybttnnum* for *Light Cavalry Battalion Number*. This mapping is not be intuitive for most users. Since users can avoid this problem, it is not a major drawback.
- **Minimize the User's Memory Load** – This is a key focus area of this research. In order to get the information necessary to build a query, the user must supply some crucial information. This research uses a wizard to get the information necessary from the user. This wizard process keeps each task to a manageable size and keeps track of the information for the user. Another important aspect of the system that contributes to this heuristic is the simplistic query visualization interface. The user interacts mainly with data points and a few simple controls. This reduces the amount of functionality a user needs to learn.
- **Be Consistent** – As discussed in Chapter 3, this heuristic is broken into two main areas. First, consistency within the system. This system implements this very well. Throughout the wizard process, all of the buttons are always in the same location. If a button is not used in a particular step, it is subdued to indicate that it



cannot be used. This makes it easy for users to know what they can and cannot do at any step in the process. Furthermore, the wizard always keeps the instructions on the left side of the screen and the user interaction items on the right side. The second aspect of this heuristic is consistency with other tools in the same domain. This research implements a visualization that is based on a familiar multi-dimensional query visualization scheme. This scheme has been tested and shown to improve user comprehension using InfoCrystal and BubbleWorld.

- **Provide Feedback** – This system provides feedback throughout the process. While the user is using the wizard, the location within the wizard is displayed at the top. Also, if the system is accessing information from the repository, the user is informed of this. This prevents the appearance that the system has locked up while performing the operation. In the query visualization, the user is provided with numerous types of feedback. First, the types that are part of the collection are displayed in the legend. This legend also shows the color assigned to each type and whether that type is displayed or not. Second, the user can determine the zoom level based on a slider. Third, when a user clicks on a data point, two more instances of feedback are provided. A shaded polygon appears graphically indicating the relevance scores in relation to each query field. Also, the metadata values for the selected data point are displayed in a table. This feedback is essential to the user, and plays a large role in comprehension.
- **Provide Clearly Marked Exits** – This system provides a standard set of navigation buttons along with tabs at the top of the display to help the user get where they want to go. If a user selects the wrong option or just wants to go back and review

his options, the navigation buttons and tabs allow him to do this. This process of navigation is very easy to accomplish.

- Provide Shortcuts – The visualization portion of this system is not very complex for the user. Therefore, there is not much need for shortcuts. However, during the wizard process, the user is presented with both the navigation buttons at the bottom and the tabs at the top. The tabs take a user directly to the step he wants to go to. The buttons at the bottom may take him through other steps first depending on which step he is on. This heuristic is applied in this system, however, it benefit is minimal.
- Provide Good Error Messages – This heuristic is a very basic, but very important concept. In this system, the heuristic is implemented very well. If a user forgets to enter some information or make a selection, the system will tell the user specifically what needs to be done to correct the situation.
- Error Prevention – Due to the simplistic nature of this user interface, this heuristic is met inherently. Since the wizard walks the user through each step of the way, there are very few places that the user can cause an error in the system.

By satisfying all nine of these heuristics to such a high degree, this visualization is definitely an effective means for gathering the required information for a query and displaying the results graphically.

### **5.1.3 Standard Query Interface**

This aspect of the system ensures that future systems are able to interface with the IR model to find needed resources. The *JBIField* object of this system satisfies this

requirement by providing an extensible framework that can be accessed to perform IR. This class also provides flexibility to the Query Wizard in creating the input fields necessary for that data type.

#### **5.1.4 Query Translation**

This requirement is set forth to allow more seamless integration of legacy systems in the future. This research does minimally satisfy this requirement by providing an interface that can be used to translate legacy system queries. However, this research does not provide any implementations of this interface, since no specific legacy systems were identified for integration.

#### **5.1.5 Query Refinement**

This requirement was also only minimally satisfied by this research. The system allows the user to revisit his query and make modifications. However, the system does not provide any mechanism to aid the user in “intelligent” query modification.

#### **5.1.6 Resource List Management**

The system did not have to directly implement this requirement. However, this research did evaluate the effectiveness of the product being used. It found that the Jini<sup>TM</sup> Lookup Service included with the CoABS environment provides excellent support for maintaining a list of resources in the system. Since this technology also existed, this system made use of it.

#### **5.1.7 Resource Management**

Like the previous requirement, the CoABS environment also satisfied this goal. Through its use of leases, CoABS is able to effectively maintain a good list of resources without overtaxing the communications medium.

## **5.2 System Impact**

Based on the discussion in the previous section, it is easy to see that this research was a success. It set out to provide a framework for querying heterogeneous systems and displaying the results graphically for the user. This system fulfilled these requirements and provides a robust tool for the user that adheres to the heuristics of a good user interface. However, the benefits of this system stretch far beyond these requirements. This system is only part of a much larger concept, the Joint Battlespace Infosphere (JBI).

With this research (and others) in place, some of the complexities of the JBI concept are mitigated, making it easier to implement. By providing an extensible way to handle IR in this heterogeneous environment, this system helps pave the way toward composable systems. One of the key difficulties in composable systems is finding the right data at the right time. This system definitely makes that possible.

This research has particular value in two key areas. The first area is found in the visualization where the system uses a shaded polygon to depict the relevance scores of a single data point. The second area is the method by which the system performs Information Retrieval on heterogeneous data.

### **5.2.1 Shaded Polygons**

During this research, it was determined that “false-positive” results could find their way into the visualization. This could happen based on the fact that a multi-dimensional query is being represented in two-dimensional space (i.e. a computer screen). Since there are only two true axes, the introduction of more axes causes conflict between them. The impact of this problem is relatively small, but can be lessened even more by using the shaded polygons presented in this research. These polygons provide

the user a synopsis of the relevance scores on demand. A user can quickly click through several possible points and find the “best” match.

### **5.2.2 Method for Heterogeneous Queries**

Another valuable outcome of this research is the method used to perform heterogeneous queries. Although metadata is not a new concept, most implementations of metadata involve only textual entries. However, this severely limits the querying power and the precision of the Information Retrieval process. Even most multi-dimensional query systems focus on a single data type. This is the case with Bubble World and InfoCrystal. Both of these systems focus solely on textual searches of documents. Another beneficial aspect of this research’s query method is the use of a structured metadata scheme that allows objects to be classified and compared correctly. This structure helps ensure compatibility between numerous systems, which enhances resource sharing.

## **5.3 Future Work**

In this section, future modifications to the system are discussed, based on the research completed during this project. Although all of these ideas surfaced during the research, there was insufficient time for implementation. If these modifications are incorporated into the system, the system will have a much more viable role in the JBI implementation.

### **5.3.1 Bubble World Incorporation**

One of the main improvements that can be made to this system is the incorporation of the Bubble World application. Unlike this system, Bubble World allows direct interaction with the query by allowing the user to add or remove “bubbles” from

query display. It also allows users to re-weight each term and change the location of each term. These enhancements would be a great addition to the system developed during this research.

However, Bubble World is tailored specifically for use only in textual searches. Modifications would have to be made that allow the user to specify different data types and provide different inputs (i.e. a range of numbers instead of a word). Also, Bubble World does not account for the error induced by opposing axes. The shaded polygon used in this research would need to be added to aid the users in selecting the correct data point. With these modifications, Bubble World could easily be used as the visualization framework for this system.

### **5.3.2 CoABS Integration**

This improvement involves two main areas. First, CoABS should be extended to provide a true middle-ware solution for the JBI concept. Currently, there are numerous “holes” in which the developer must use Jini<sup>TM</sup> resources to accomplish what needs to be done. This should not be the case. If CoABS were extended to cover these “holes”, it would provide a more complete solution. For example, the system developed during this research would not function anymore if CoABS transitioned to using a tool other than Jini<sup>TM</sup> to implement its functionality, due to the numerous calls to Jini<sup>TM</sup> resources. However, if all of the calls could be abstracted through the CoABS layer, it would not matter what CoABS used.

The second enhancement of CoABS involves integrating this system as a core resource that is provided with all versions of the CoABS grid. This would ensure that all

developers and users of the JBI would have access to this tool. This wide dissemination is necessary to ensure the success of a JBI implementation.

### **5.3.3 Better Information Retrieval**

This research is concerned mainly with the proof of the concept that heterogeneous data sources can be queried simultaneously. To test this, the system implements a standard extended boolean Information Retrieval model for text and a simplistic range calculation for numbers. While these models work for proving the concept, better models should be tested and applied before this system is used for actual queries. Due to the system design, this is a very straightforward issue and can be integrated with the existing system almost effortlessly.

### **5.3.4 Query Translators**

This research only developed an interface for including query translators in the system; however, no actual implementations are provided. These translators are unique to each legacy query standard such as SQL. Even within each standard, there may be a need for different translators depending on the systems that use the translators to access the resources.

### **5.3.5 Query Refinement**

As mentioned in previous sections, this research did not provide a very robust means for the user to refine his query. While the mechanism is there, the only thing a user can do is to revisit his query and change some parameters. If more advances are made in this area, the system would have the potential to provide much more accurate results for the user's query.

### 5.3.6 Hierarchical Template List

In the Query Wizard and the Publication Wizard, the list of metadata templates that the user is shown does not depict any of the inheritance that is found in the metadata structure. By adding this hierarchy to the list, it would be easier for the user to determine at what level he needs to query. If he sees that *Missiles* and *Shells* both inherit from *Munitions*, and he wanted to include both results in the query, he would know to select *Munitions* from the list. Without this hierarchy, it is impossible to determine (other than by common sense) the chain of inheritance.

## 5.4 Conclusion

This research focused on two main problem areas: Information Retrieval in heterogeneous systems and Visualization of multi-variable queries. These two areas are tied together in the application of the JBI. Due to its vast size and heterogeneous nature, this research is vital in creating a viable means to locate resources within it. The system produced during this research aids in understanding both the problem domain and possible solutions to the problem. It incorporates many accepted techniques in these two areas while also proposing new techniques to aid in user comprehension. These new techniques should be incorporated into any implementation of the JBI to provide a feasible means of managing the vast amounts of resources contained in a JBI.



## Bibliography

1. Boyd, John R., "The Essence of Winning and Losing", PowerPoint Briefing, 28 June 1995.
2. Card, Stuart K., J. D. MacKinlay, Ben Shneiderman, "Information Visualization", Readings in Information Visualization: Using Vision to Think, pg 1-34, Academic Press, San Diego, CA 1999.
3. Cody, W. F., and others, "Querying Multimedia Data from Multiple Repositories by Content: the Garlic1 Project" citeseer, pg 420-439  
<http://citeseer.nj.nec.com/cache/papers/cs/420>, 2001.
4. Darpa Website "What is CoABS" <http://coabs.globalinfotek.com/> 2002
5. Department of Defense. Joint Vision 2020. Chairman Joint Chief of Staff.  
<http://www.dtic.il/jcs/>. 20 September, 2001.
6. Dublin Core Metadata Initiative Website. <http://www.dublincore.org/>
7. Jennings, Nicholas, "On agent-based software engineering" Artificial Intelligence
8. Kahn, Martha and others "DARPA CoABS Grid Users Manual" version 3.3.0, February 2002.
9. McCarthy, J. and other. Building the Joint Battlespace Infoshpere, Volume 1: Summary. Research Report SAB-TR-99-01, USAF Scientific Advisory Board, December 17, 1999.
10. Roell, Richard J. A Data Framework for Integrating Heterogeneous Systems Using Agents, XML AND CoABS. Masters Thesis. Department of Computer Systems, Air Force Institute of Technology, Wright Patterson Air Force Base, OH, March 2003.
11. Spoerri, Anselm, "InfoCrystal: A Visual Tool for Information Retrieval", Readings in Information Visualization: Using Vision to Think, pg 140-147, Academic Press, San Diego, CA 1999.
12. Stang, Mark and Stephen Whinston, "Enterprise Computing with Jini Technology." IT Professional, Vol. 3 No. 1: 33-38 (January/February 2001)
13. Sun Microsystems "Jini Network Technology Overview"  
<http://www.sun.com/software/jini/overview/index.html> (1994)
14. Sun Microsystems "Jini Network Technology" 2001 Palo Alto, CA  
<http://www.sun.com/software/jini/whitepapers/jini-datasheet0601.pdf>
15. Tannenbaum, A. and M. Steen, Distributed Systems, Principles and Paradigms. Prentice Hall, Upper Saddle River, NJ 2002

16. United States Air Force Scientific Advisory Board. "Information Management to Support the Warrior." Report SAB-TR-98-02, December 1998.
17. United States Air Force Scientific Advisory Board. "Building the Joint BattleSpace Infosphere." SAB-TR-99-02. December 17, 1999.
18. Van Berendonck, Chris, Bubble World - A Novel Visual Information Retrieval Technique, Masters Thesis. Department of Computer Systems, Air Force Institute of Technology, Wright Patterson Air Force Base, OH, March 2002.
19. Yates, Ricardo, Neto, Berthier, Modern Information Retrieval. Addison Wesley, Harlow, England, 1999
20. Molich, Rolf, and Jakob Nielsen, "Improving a human-computer dialogue". *Communications of the ACM*. New York, NY. 1990. 338-348.
21. Savage, Pamela. "User Interface Evaluation in an Iterative Design Process: A Comparison of Three Techniques". *CHI '96 Companion*, Vancouver, BC Canada. 1996. 307-308.
22. Kahn et al. "DARPA CoABS Grid Users Manual"  
<http://coabs.globalinfotek.com/public/downloads/Grid/documents/GridUsersManual.v4-draft.doc>. October 2002.

<b>REPORT DOCUMENTATION PAGE</b>				Form Approved OMB No. 074-0188	
<p>The public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of the collection of information, including suggestions for reducing this burden to Department of Defense, Washington Headquarters Services, Directorate for Information Operations and Reports (0704-0188), 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302. Respondents should be aware that notwithstanding any other provision of law, no person shall be subject to an penalty for failing to comply with a collection of information if it does not display a currently valid OMB control number.</p> <p><b>PLEASE DO NOT RETURN YOUR FORM TO THE ABOVE ADDRESS.</b></p>					
1. REPORT DATE (DD-MM-YYYY) 25-03-2003		2. REPORT TYPE Master's Thesis		3. DATES COVERED (From – To) Aug 2002 – Mar 2003	
4. TITLE AND SUBTITLE  IMPLEMENTING AN INFORMATION RETRIEVAL AND VISUALIZATION FRAMEWORK FOR HETEROGENEOUS DATA TYPES				5a. CONTRACT NUMBER	
				5b. GRANT NUMBER	
				5c. PROGRAM ELEMENT NUMBER	
6. AUTHOR(S)  Kowalchuk, Andrew J., Captain, USAF				5d. PROJECT NUMBER	
				5e. TASK NUMBER	
				5f. WORK UNIT NUMBER	
7. PERFORMING ORGANIZATION NAMES(S) AND ADDRESS(S) Air Force Institute of Technology Graduate School of Engineering and Management (AFIT/EN) 2950 P Street, Building 640 WPAFB OH 45433-7765				8. PERFORMING ORGANIZATION REPORT NUMBER  AFIT/GCS/ENG/03-09	
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES) Rome Labs AFRL Attn: Mr. Vaughn Combs Air Force Material Command (AFMC) 525 Brooks Road DSN: 587-7282 Rome, NY 13441 e-mail: VAUGHN.COMBS@RL.AF.MIL				10. SPONSOR/MONITOR'S ACRONYM(S)	
				11. SPONSOR/MONITOR'S REPORT NUMBER(S)	
12. DISTRIBUTION/AVAILABILITY STATEMENT  APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED.					
13. SUPPLEMENTARY NOTES					
14. ABSTRACT In today's information focused world, there is no lack of entities focused on information gathering. However, there is still a widespread epidemic of information starvation in the Department of Defense (DoD). This starvation is attributed to the lack of interoperability between information gatherers and information consumers. To alleviate this problem, the DoD has put forth a vision of a Joint Battlespace Infosphere (JBI). This research proposes a framework for sharing and finding resources in a JBI. The framework uses an extensible metadata specification, agent technology, and the Control of Agent Based Systems (CoABS). It provides several tools for publication and subscription of resources, including a visual query wizard and a visualization of the results. This framework and tools provide visual query capability for the heterogeneous resources within the JBI.					
15. SUBJECT TERMS Control of Agent Based Systems, Joint Battlespace Infosphere, Agents, Metadata, Heterogeneous Systems, Multi-dimensional Query, Multi-Media Query, C2 Systems					
16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			Timothy M. Jacobs, Lt Col, USAF (ENG)
U	U	U	UU	83	19b. TELEPHONE NUMBER (Include area code) (937) 255-6565, ext 4279; e-mail: Timothy.Jacobs@afit.edu